AFRL-RI-RS-TR-2007-215
In-House Final Technical Report
October 2007

DEVELOPMENT AND EVALUATION OF FUSION TECHNIQUES

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

NOTICE AND SIGNATURE PAGE

AFRL-RI-RS-TR-2007-215 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/	/s/
JON JONES, Chief Fusion Technology Branch Information & Intelligence Exploitation Division	JOSEPH CAMERA, Chief Information & Intelligence Exploitation Division Information Directorate

REPORT DOCUMENTATION PAGE	*Form Approved* **OMB No. 0704-0188**

1. REPORT DATE *(DD-MM-YYYY)* OCT 2007	**2. REPORT TYPE** Final	**3. DATES COVERED** *(From - To)* Jul 05 – Jun 07

4. TITLE AND SUBTITLE DEVELOPMENT AND EVALUATION OF FUSION TECHNIQUES	**5a. CONTRACT NUMBER** In-House
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER 62702F
6. AUTHOR(S) Mark G. Alford, Maria L. Scalzo, Adnan Bubalo and Eric C. Jones	**5d. PROJECT NUMBER** 459E
	5e. TASK NUMBER DE
	5f. WORK UNIT NUMBER FT
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) AFRL/RIEA 525 Brooks Rd Rome NY 13441-4505	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/RIEA 525 Brooks Rd Rome NY 13441-4505	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSORING/MONITORING AGENCY REPORT NUMBER AFRL-RI-RS-TR-2007-215

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA# WPAFB 07-0033

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The purpose of this effort was to investigate and develop promising innovative technologies that hold promise for improvements in the Air Force target tracking and fusion capabilities. The problem statement is that Nonlinear Non-Gaussian Processes (NNGP) present a major challenge in all types of military problems. This is because the real-world is nonlinear and non-Gaussian, despite assumptions made in most conventional fusion algorithms. This in-house program addressed this problem by researching and developing tracking filters that do not presume a linear Gaussian world. The most famous of these is the Particle Filter (PF). Progress made includes development of a MATLAB PF simulation capability for in-house analysis/testing and preliminary investigation of PF capabilities. Prior to this effort AFRL/IF had no capability in this area and PF were not being investigated. DEFT developed a two-dimensional PF and performed Monte Carlo simulation runs to test the performance of the PF as compared to the Extended Kalman Filter (EKF) for multiple bearings only sensors (ESM sensors). Results showed that in certain cases, the Monte Carlo averaged tracking error variance of the PF was much better than that of the EKF, but with ten times the computational cost. Under these same conditions (50 particles, 100 Monte Carlo runs, two bearings only sensors), the EKF lost track 28% of the time whereas the PF did not lose track at all. This effort established future directions for research. There is a need to do more nonlinear filtering analysis, development and enhancements under realistic scenarios with varying degrees of nonlinearity. Considerable study is needed to fully determine the conditions under which these techniques will lead to improved performance. It is possible that a system could be developed employing multiple nonlinear filters.

15. SUBJECT TERMS
Nonlinear filtering, target tracking, particle filtering, Electronic Support Measures, bearings only tracking

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Mark G. Alford
a. REPORT U	**b. ABSTRACT** U	**c. THIS PAGE** U	UL	94	**19b. TELEPHONE NUMBER (*Include area code)*** N/A

Table of Contents

List of Figures

1.0 Introduction

The Development and Evaluation of Fusion Techniques (DEFT) in-house program was initiated with the intent of addressing multiple areas. The first purpose of DEFT was to extend the development of algorithmic concepts that had been identified and explored under the previous Complementary Advanced Fusion Exploration (CAFÉ) in-house program. CAFÉ investigated promising recent innovations, primarily involving mathematical concepts within the area of basic research, and did a preliminary assessment of their validity as well as a preliminary qualitative assessment of each concept's potential for application to Air Force Command, Control, Intelligence, Reconnaissance, and Surveillance (C2ISR) capabilities and systems. DEFT was intended to continue the investigation of potential exploratory development applications of the CAFÉ algorithmic concepts and to investigate concepts that were revealed as extensions of the original CAFÉ concepts.

Areas of investigation under CAFÉ that were originally identified for further exploration under DEFT included the use of Trilinear tensors (including Homography) for use in the reprojection of images, the Nash (Grocholsky) Equilibrium Approach to Sensor Management, the Virtanen Methodology Approach, improvements in handling the Out of Sequence Problem (OOSP) and Out of Sequence Measurement (OOSM) updating, and further study in the area of Particle Filters and their applications.

In addition, DEFT was intended to study other Exploratory Development topics not previously investigated under CAFÉ. These topics included Bayesian Network exploitation for fusion, Optimization of ISR platform route determination, and Radio Frequency (RF) tag utilization.

The first area to be pursued under DEFT was the area of Particle Filtering. As work proceeded on this area, it became clear that study in this area should become the focus of the DEFT in-house program. This is what took place. As a result, no work took place on the other originally intended topics.

As the work focused on Particle Filtering, the directions of the work were clarified. The nature of the problem being addressed was that traditional trackers, based around the Kalman Filter, were designed to deal with linear targets with Gaussian noise statistics. However, targets are not constrained to behave in these ways, and so methods are needed to deal with targets that are not behaving in these ways.

The Particle Filter is one such method. It does not make the assumptions of linearity and Gaussianity that the Kalman Filter does, but it requires a large number of computations, depending on how parameters are set.

To assess the usefulness of the Particle Filter, a systematic program was initiated to investigate it to assess its practicality and usefulness. This program involved a number of areas. First, further study was required to understand the mathematical basis for why the Particle Filter provides improved filtering and is a basis of improved tracking in situations of interest. Next, it was desired to create and/or obtain a simulation as a framework for systematic parametric testing of the performance of the particle filter versus other types of filters. However, the goals of this effort, in addition to those of CAFÉ were to take a more focused approach to fusion techniques. That is, DEFT looked to focus on a deep mathematical understanding of algorithms researched under CAFÉ, as well as to go beyond the work done under CAFÉ by creating in-house implementations for the testing of these algorithms. The team looked to create an in-house testbed that would serve as a model for other multi-sensor fusion studies, as well as to, as the name suggests, evaluate the performance of these methods. These tasks were created to advance and assess new algorithmic concepts in order to enhance the capabilities available to track maneuvering and unpredictable targets. DEFT searched for algorithms and techniques that would optimally fuse the information available.

The scenarios for concern included those situations that involve highly non-linear and non-Gaussian characteristics. While existing methods, such as the Kalman (KF) and Extended Kalman Filters (EKF), perform sufficiently in linear and Gaussian cases, these algorithms tend to lose tracks and perform unreliably when faced with data that exhibits alternative behavior. In order to gain increased tracking accuracy and reliability, the team proposed the prospects of using a Particle Filtering (PF) algorithm. Based on research done under CAFÉ, as well as the fact that there are no assumptions of linearity or

Gaussianity made in the PF, these filters showed potential for providing an innovative approach to the non-linear and non-Gaussian scenarios under which the Kalman Filters have shown possible weaknesses.

The approach of this effort was to mathematically characterize the algorithms under investigation. Once a working definition and understanding of the techniques was developed, the team, in collaboration with Syracuse University, created an implementation of these algorithms to evaluate the performance of the techniques on the standards of position error, computational expenditures, and characterizations of situations under which a filter showed improved tracking results.

2.0 Kalman Filtering

Developed in the 1960's, the Kalman Filter (KF) was the first of a family of algorithms aimed at estimating the current state of an object. The KF is an optimal (unbiased) estimator for objects with linear and Gaussian characteristics, for this algorithm assumes that the object has these attributes. The basic theory is to predict the current state of an object, upon reception of observations, by taking the weighted sum of the estimate and the most recent observations. Hence, as the general theory suggests, the algorithm is broken up into prediction and update stages.

2.1 Algorithm

Prediction

The equations which calculate the predicted state vector and covariance matrix are:

$$\tilde{\mathbf{x}} = F \cdot \hat{\mathbf{x}} + \mathbf{OWN} \tag{1}$$

$$\tilde{\mathbf{P}} = F \cdot \hat{\mathbf{P}} \cdot F^T + \mathbf{Q} \tag{2}$$

- $\hat{\mathbf{x}}$ and $\tilde{\mathbf{x}}$ are the state vector

- $\hat{\mathbf{P}}$ and $\tilde{\mathbf{P}}$ are error covariance
- F is the state transition matrix
- $\mathbf{Q}$ is process noise covariance
- **OWN** is a control vector

Note that when a matrix or vector symbol is covered with a hat, the parameter represents a measurement-updated *estimate*; when a matrix or vector symbol is covered with a tilde, this represents a parameter that is a time-updated *prediction*. Therefore the initial state estimate will be represented by $\hat{\mathbf{x}}$ and the initial error covariance matrix will be represented by $\hat{\mathbf{P}}$.

As a recursive algorithm, the process must start with an initially defined state estimate and error covariance. The elements of the state vector are the parameters that are to be estimated. While the state vector could include a wide range of components, when tracking ground targets the parameters of interest are typically the position and velocity of the object in Cartesian coordinates. A depiction of the state vector may appear as below.

$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ \vdots \end{bmatrix}, \qquad (3)$$

A dot above a variable indicates the first derivative of motion in that direction (also called the velocity in that direction). Now, the error covariance matrix provides information about the accuracy of the state estimate, or can be viewed as a measure of how the components of the state vector vary with respect to each other. In a more mathematically thorough statement, according to the definition of state error covariance, the error covariance matrix will be an ordered set of the covariance between each ordered pair of differences among the actual state and the estimated components of the state vector.

$$\hat{\mathbf{P}} = E[(x - \hat{x})(x - \hat{x})^T] \qquad (4)$$

$$\widetilde{\mathbf{P}} = E[(x - \widetilde{x})(x - \widetilde{x})^T] \qquad (5)$$

Similarly, the process noise covariance matrix, $\mathbf{Q}$, takes into account the error involved in the model that is chosen for the system. This error can be incorporated into the system, by making assumptions concerning the models chosen for the system, or as will be observed in the following section, through an approximation of the system parameters that does not include all higher order derivatives.

The state transition matrix F is essential to understanding the Kalman Filter. This is the matrix that, when multiplied by the state vector, will produce the predicted state of the object at the subsequent time step. Since, this research is primarily concerned with tracking and motion, the matrix that will be used most often in this paper is easily derived from the common kinematic equations found in physics. Yet, it should be noted that the KF can be applied to other applications, and therefore may have other appropriate transition functions. The equations that will be used are as follows:

$$x_k = x_{k-1} + \dot{x}_{k-1} \cdot dt + 1/2\ddot{x}_{k-1} \cdot dt^2 ... \qquad (5)$$

$$\dot{x}_k = \dot{x}_{k-1} + \ddot{x}_{k-1} \cdot dt..., \qquad (6)$$

The ellipses indicate the remaining derivatives of motion, with which we are not concerned due to our linear assumption. Taking the coefficients of the elements of state vector, namely the coefficients of velocity and position, the state transition matrix can be derived from these equations as:

$$F = \begin{bmatrix} 1 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & dt \\ 0 & 0 & 0 & 1 \end{bmatrix}. \qquad (7)$$

The ownship or control vector, which adjusts the prediction for a sensor mounted on a moving platform (such as on an aircraft or a car), is optional. Although some refer to this as the multiplication of a transition matrix $\mathbf{B}$ and a state vector $\mathbf{u}$ with components of the ownship's velocity and acceleration which are assumed to be known, here it will be referred to as **OWN**. As described, the **OWN** is often represented as:

$$\mathbf{B} \cdot \mathbf{u} = \begin{bmatrix} dt & \frac{dt^2}{2} & \cdots \\ 1 & dt & \\ \vdots & & \ddots \end{bmatrix} \cdot \begin{bmatrix} velocity \\ acceleration \\ \vdots \end{bmatrix} \qquad (8)$$

Update

The update, upon the reception of a measurement, corrects the time updated estimate and error covariance. These formulas are derived from the minimization of the MSE of the state and the measurement, making the Kalman Filter an optimal estimator. The update formulas are as follows:

$$\mathbf{K} = \widetilde{\mathbf{P}} \cdot \mathbf{H}^T [\mathbf{H} \cdot \widetilde{\mathbf{P}} \cdot \mathbf{H}^T + \mathbf{R}]^{-1} \qquad (9)$$

$$\hat{\mathbf{P}} = (\mathbf{I} - \mathbf{K} \cdot \mathbf{H})\widetilde{\mathbf{P}} \qquad (10)$$

$$\hat{\mathbf{x}} = \widetilde{\mathbf{x}} + \mathbf{K}(\mathbf{z} - \mathbf{H} \cdot \widetilde{\mathbf{x}}) \qquad (11)$$

- $\mathbf{K}$ is the Kalman gain
- $\hat{\mathbf{P}}$ and $\widetilde{\mathbf{P}}$ are error covariance

- $\mathbf{H}$ is the state to measurement transformation matrix
- $\mathbf{z}$ is the measurement vector
- $\mathbf{R}$ is measurement noise covariance
- $\hat{\mathbf{x}}$ and $\tilde{\mathbf{x}}$ are the state vector

Within the update stage, the measurement-updated estimate, which is used as an input for the next iteration of the filter, is calculated by taking the sum of the time-updated estimate and a weighted difference known as the innovation. The innovation, also called the residual, is the difference between the measurement $\mathbf{z}$ and the product $\mathbf{H} \cdot \tilde{\mathbf{x}}$. The matrix $\mathbf{H}$ is necessary because it transforms a vector from the state-space to the measurement-space. Otherwise these vectors will likely not be the same size or be in the same scale. Given information from N sensors centralized measurement fusion is performed in the KF as:

$$H_k = [H_{k,S_1}^T, H_{k,S_2}^T, \ldots, H_{k,S_N}^T]^T$$

$$R_k = \begin{bmatrix} R_{k,S_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & R_{k,S_N} \end{bmatrix}$$

where $S_{1,\ldots,}S_N$ indicates the sensor from which information is related and R_k is the measurement covariance matrix.

The weight that is calculated is known as the Kalman gain, which takes into account the variance of error in the estimate, $\tilde{\mathbf{P}}$, and the variance of noise in the measurement, $\mathbf{R}$. Notice, as the estimate error covariance goes to zero, the Kalman gain goes to zero, indicating that the estimate is highly reliable. This will result in a $\hat{\mathbf{P}}$ with similarly smaller values as $\tilde{\mathbf{P}}$ and a measurement-updated estimate that is very close to the value of the time-updated estimate. Alternatively, if the measurement noise covariance approaches zero, the Kalman gain gradually progresses towards the value $\mathbf{H}^T$. As a result, the innovation will be more highly weighted and the measurement-updated estimate will be farther away from the time-updated estimate. Notice that still in this case where the Kalman gain is larger, the error covariance is still being minimized. Hence, the general idea of the Kalman gain equation is that it designates how much the innovation will change the predicted state.

2.2 Derivation

Although, we have preliminary described a few of the parameters that will be used, for this derivation some of the terms will be defined as follows:

i.	*Corrected Estimate:*	$\hat{x} = x(k \mid k)$
ii.	*Prediction:*	$\tilde{x} = x(k \mid k-1)$
iii.	*Measurement:*	z
iv.	*Weight of Prediction:*	K_1
v.	*Weight of Measurement:*	K_2

The corrected estimate can be written as the sum of the weighted prediction and the weighted measurement.

$$\boxed{\hat{x} = K_1\tilde{x} + K_2 z} \tag{1}$$

In order to proceed, it must be observed that:

$$\hat{x} = x + \hat{\varepsilon} \qquad \tilde{x} = x + \tilde{\varepsilon}$$

$$z = Hx + v$$

Note that $\hat{\varepsilon}$ and $\tilde{\varepsilon}$ are estimation errors with zero means, while z is being linearized by an H matrix with additive, zero mean noise. Hence, substituting the above equations into equation (1), we get:

$$\boxed{x + \hat{\varepsilon} = K_1(x + \tilde{\varepsilon}) + K_2(Hx + v)} \tag{2}$$

Performing some algebraic manipulation on equation (2) yields:

$$\hat{\varepsilon} = K_1(x + \tilde{\varepsilon}) + K_2(Hx + v) - x$$

$$= K_1 x + K_1\tilde{\varepsilon} + K_2 Hx + K_2 v - x \quad \text{(Distributive Property)}$$

$$= (K_1 x + K_2 Hx - x) + K_1\tilde{\varepsilon} + K_2 v \quad \text{(Commutative Property)}$$

$$= (K_1 + K_2 H - I)x + K_1\tilde{\varepsilon} + K_2 v \quad \text{(Distributive Property)}$$

$$\boxed{\hat{\varepsilon} = (K_1 + K_2 H - I)x + K_1\tilde{\varepsilon} + K_2 v} \tag{3}$$

Recall, the means of all the errors are zero or in other words:

$$E[\hat{\varepsilon}] = E[\tilde{\varepsilon}] = E[v] = 0$$

Therefore, taking the means of both sides of equation (3) gets:

$$E[(K_1 + K_2H - I)x] = 0$$
$$E[(K_1 + K_2H - I)]E[x] = 0$$
$$(K_1 + K_2H - I)E[x] = 0$$

However, since $E[x] = x$ the above implies:

$$K_1 + K_2H - I = 0$$

Solving for K_1 , it is clear that:

$$\boxed{K_1 = I - K_2H} \qquad (4)$$

Let $K_2 = K$, where K is the Kalman gain. Then, it is apparent that substituting equation (4) into equation (1) yields:

$$\hat{x} = (I - KH)\tilde{x} + Kz$$
$$\hat{x} = \tilde{x} - KH\tilde{x} + Kz$$
$$\hat{x} = \tilde{x} + Kz - KH\tilde{x}$$

$$\boxed{\hat{x} = \tilde{x} + K(z - H\tilde{x})} \qquad (5)$$

Hence, equation (5) is known as the state correction equation, which corrects the prediction $\tilde{x}$ by weighting the measurement in a way that minimizes the mean squared error. In order to implement equation (5), it is required that K be expressed in known or measurable terms. In order to do this we must, introduce the error covariance correction:

$$\hat{P} = E[\hat{\varepsilon}\hat{\varepsilon}^T]$$

In other words, the error covariance correction is the mean of the product of the estimation error and the transpose of the estimation error.

Recall that $\hat{\varepsilon}$ and $\tilde{\varepsilon}$ are estimation errors with zero means, where:

$$\hat{x} = x + \hat{\varepsilon}$$

$$\tilde{x} = x + \tilde{\varepsilon}$$

Substituting these equations into (5):

$$x + \hat{\varepsilon} = (I - KH)(x + \tilde{\varepsilon}) + Kz$$

Now, recall that $z = Hx + v$, yielding:

$$\begin{aligned}
x + \hat{\varepsilon} &= (I - KH)(x + \tilde{\varepsilon}) + K(Hx + v) \\
x + \hat{\varepsilon} &= (I - KH)x + (I - KH)\tilde{\varepsilon} + KHx + Kv \\
x + \hat{\varepsilon} &= x - KHx + (I - KH)\tilde{\varepsilon} + KHx + Kv \\
\hat{\varepsilon} &= (I - KH)\tilde{\varepsilon} + Kv
\end{aligned}$$

From the definition of $\hat{P}$:

$$\hat{P} = E\{([I - KH]\tilde{\varepsilon} + Kv)(\tilde{\varepsilon}^T [I - KH]^T + v^T K^T)\}$$

$$\hat{P} = E\{(I - KH)\tilde{\varepsilon}\tilde{\varepsilon}^T (I - KH)^T + (I - KH)\tilde{\varepsilon}v^T K^T + Kv\tilde{\varepsilon}^T (I - KH)^T + Kvv^T K^T\}$$

$$\hat{P} = E\{(I - KH)\tilde{\varepsilon}\tilde{\varepsilon}^T (I - KH)^T\} + E\{(I - KH)\tilde{\varepsilon}v^T K^T\} + E\{Kv\tilde{\varepsilon}^T (I - KH)^T\} + E\{Kvv^T K^T\}$$

Since the estimation error, $\tilde{\varepsilon}$, is zero mean, and v is zero mean non-Gaussian white noise:

$$E\{\tilde{\varepsilon}v^T\} = E\{v\tilde{\varepsilon}^T\} = 0$$

Hence, part of the sum cancels, resulting in:

$$\hat{P} = (I - KH) * E\{\tilde{\varepsilon}\tilde{\varepsilon}^T\} * (I - KH)^T + K * E\{vv^T\} * K^T \qquad (6)$$

As previously defined:

$$\tilde{P} = E\{\tilde{\varepsilon}\tilde{\varepsilon}^T\}$$

$$R = E\{vv^T\}$$

Therefore, equation (6) becomes:

$$\boxed{\hat{P} = (I - KH)\tilde{P}(I - KH)^T + KRK^T} \qquad (7)$$

Performing some algebraic manipulation as follows:

$$\hat{P} = (\tilde{P} - KH\tilde{P})(I - H^T K^T) + KRK^T$$
$$\hat{P} = \tilde{P} - KH\tilde{P} - \tilde{P}H^T K^T + KH\tilde{P}H^T K^T + KRK^T$$

Now, taking the trace of each side:

$$tr(\hat{P}) = tr(\tilde{P} - KH\tilde{P} - \tilde{P}H^T K^T + KH\tilde{P}H^T K^T + KRK^T) \qquad (8)$$

This enables us to combine some terms due to the fact that:

$$tr(KH\widetilde{P}) = tr(\widetilde{P}H^T K^T)$$

Hence, equation (8) is equivalent to:

$$tr(\hat{P}) = tr(\widetilde{P} - 2KH\widetilde{P} + KH\widetilde{P}H^T K^T + KRK^T) \qquad (9)$$

In order to simplify this equation further, we can use the fact that:

$$\frac{d}{dK} tr(\widetilde{P}) = 0$$

Therefore, an approach is to take the derivative with respect to K of both sides of equation (9):

$$\frac{d}{dK} tr(\hat{P}) = \frac{d}{dK} tr(KH\widetilde{P}H^T K^T - 2KH\widetilde{P} + KRK^T)$$

$$\frac{d}{dK} tr(\hat{P}) = \frac{d}{dK} tr(KH\widetilde{P}H^T K^T) - \frac{d}{dK}(2KH\widetilde{P}) + \frac{d}{dK}(KRK^T) \qquad (10)$$

For book keeping label the parts of the above equation as:

(A) $\frac{d}{dK} tr(KH\widetilde{P}H^T K^T)$

(B) $\frac{d}{dK} tr(2KH\widetilde{P})$

(C) $\frac{d}{dK} tr(KRK^T)$

Now, note the following properties of $\frac{d}{dX} tr$:

(i) $\frac{d}{dX} tr(XYX^T) = 2XY$

(ii) $\frac{d}{dX} tr(YXZ) = Y^T Z^T$

By property (i) equations (A) becomes:

$$\frac{d}{dK}tr(KRK^T) = 2KR \qquad \text{, where } X \text{ is taken as } K \text{ and } Y \text{ is taken as } R$$

Similarly (C) becomes:

$$\frac{d}{dK}tr(KH\tilde{P}H^T K^T) = 2KH\tilde{P}H^T \text{ , where } X \text{ is taken as } K \text{ and } Y \text{ is taken as } HPH^T$$

By property (ii) and the property $tr(xx^T) = tr(x^T x)$ equations (B) is:

$$\frac{d}{dK}tr(2KH\tilde{P}) = \frac{d}{dK}(2\tilde{P}^T KH)$$

$$\frac{d}{dK}(2\tilde{P}^T KH) = 2\tilde{P}H^T \text{ , where } Y \text{ is taken as } P^T \text{ and } Z \text{ is taken as } H$$

We now have all the terms, transforming equation (10) into:

$$\frac{d}{dK}tr(\hat{P}) = 2KH\tilde{P}H^T - 2\tilde{P}H^T + 2KR \tag{11}$$

Since $tr(\hat{P})$ is independent of K:

$$\frac{d}{dK}tr(\hat{P}) = 0$$

Hence, equation (11) results in:

$$2KH\tilde{P}H^T - 2\tilde{P}H^T + 2KR = 0 \tag{12}$$

Equation (12) can be manipulated to finally yield K, the Kalman Gain:

$$2KH\tilde{P}H^T - 2\tilde{P}H^T + 2KR = 0$$
$$2K(H\tilde{P}H^T + R) - 2\tilde{P}H^T = 0$$
$$K(H\tilde{P}H^T + R) = \tilde{P}H^T$$

$$\boxed{K = \tilde{P}H^T(H\tilde{P}H^T + R)^{-1}} \tag{13}$$

Now, let:

$$S = H\tilde{P}H^T + R \tag{14}$$

Then the Kalman Gain is:

$$K = \tilde{P}H^T S^{-1} \tag{15}$$

Now, we will go back to equation (7) in order to express $\hat{P}$ solely in terms of K, H, $\tilde{P}$, and I (the identity matrix). Recall equation (7) is:

$$\hat{P} = (I - KH)\tilde{P}(I - KH)^T + KRK^T$$

Using equations (14) and (15), as well as applying some algebraic manipulation:

$$\begin{aligned}
\hat{P} &= (\tilde{P} - KH\tilde{P})(I - KH)^T + KRK^T \\
&= \tilde{P} - KH\tilde{P} - \tilde{P}H^T K^T + KH\tilde{P}H^T K^T + KRK^T \\
&= \tilde{P} - KH\tilde{P} - \tilde{P}H^T K^T + K(H\tilde{P}H^T K^T + RK^T) \\
&= \tilde{P} - KH\tilde{P} - \tilde{P}H^T K^T + K(H\tilde{P}H^T + R)K^T \\
&= \tilde{P} - KH\tilde{P} - \tilde{P}H^T K^T + KSK^T \\
&= \tilde{P} - \tilde{P}H^T S^{-1}H\tilde{P} - \tilde{P}H^T(\tilde{P}H^T S^{-1})^T + \tilde{P}H^T S^{-1}SS^{-1}H\tilde{P} \qquad (16)
\end{aligned}$$

Make note that:

$$\tilde{P} = \tilde{P}^T$$

$$(S^{-1})^T = S^{-1}$$

Then upon further simplification of (16), the error covariance correction equation results as follows:

$$\begin{aligned}
\hat{P} &= \tilde{P} - \tilde{P}H^T S^{-1}H\tilde{P} - \tilde{P}H^T S^{-1}H\tilde{P} + KH\tilde{P} \\
&= \tilde{P} - 2\tilde{P}H^T S^{-1}H\tilde{P} + KH\tilde{P} \\
&= \tilde{P} - 2KH\tilde{P} + KH\tilde{P} \\
&= \tilde{P} - KH\tilde{P}
\end{aligned}$$

$$\boxed{\hat{P} = (I - KH)\tilde{P}} \tag{17}$$

Now, all that is left is $\tilde{P}$, which can be derived from the following set of equations:

(i) $\tilde{x} = \phi\hat{x}$

(ii) $x_{k+1} = \phi x_k + w$

(ii) $\tilde{\varepsilon} = x_{k+1} - \tilde{x}$

The above introduces a new matrix phi, which is the state transition matrix, and w, which is a zero mean white noise. Applying equation (i) and (ii) in equation (iii) yields:

$$\begin{aligned}
\tilde{\varepsilon} &= (\phi x_k + w) - \phi\hat{x} \\
&= \phi(x_k - \hat{x}) + w \\
&= \phi\hat{\varepsilon} + w
\end{aligned}$$

Recall that the error covariance is the mean of the estimation error and its transpose. Hence, the following is the error covariance equation:

$$\tilde{P} = E\{\tilde{\varepsilon}\tilde{\varepsilon}^T\}$$
$$= E\{(\phi\hat{\varepsilon} + w)(\phi\hat{\varepsilon} + w)^T\}$$
$$\boxed{\tilde{P} = \phi\hat{P}\phi^T + Q} \qquad (18)$$

Finally, the prediction equation after solution to differential equations assuming zero-mean, white process is:

$$\boxed{\tilde{x} = \phi\hat{x} + \int \beta\mu dt + w} \qquad (19)$$

In summary, the Kalman Filter equations are:

(I) $\tilde{x} = \phi\hat{x} + \int \beta\mu dt + w$

(II) $\tilde{P} = \phi\hat{P}\phi^T + Q$

(III) $K = \tilde{P}H^T(H\tilde{P}H^T + R)^{-1}$

(IV) $\hat{P} = (I - KH)\tilde{P}$

(V) $\hat{x} = \tilde{x} + K(z - H\tilde{x})$

(VI) $z = Hx + v$

2.3 Evaluation

As discussed previously, the Kalman filter is an optimal filter for linear-Gaussian tracking. This technique does not require much space in memory since only the results of the previous time step are necessary for computation. The computation itself is not complicated, so results are achieved rapidly. It therefore finds extensive uses for tracking vehicles along roadway systems, or distance-oriented flight paths.

However, the Kalman Filter is not applicable to nonlinear situations such as in close combat or for highly noisy measurements which might occur in cluttered environments, such as tracking ground transportation in cities. Therefore other techniques need to be utilized for these situations.

3.0 The Extended Kalman Filter

3.1 Theory

An adaptation to the Kalman Filter that addresses nonlinear situations is the Extended Kalman Filter. This extension uses the same basic premise as the KF, but it assumes the local linearity of the track based on the definition of derivative in order to prevent divergence when attempting to track highly nonlinear paths. This technique allows for more flexibility in its application. Although it is not an optimal estimator, it does approximate the state to the first two moments of its Taylor series expansion.

3.2 Algorithm

The Extended Kalman Filter uses the same equations as the original Kalman filter, replacing the state transition matrix and the state-to-measurement transformation matrix with their respective Jacobian matrices. These Jacobian matrices are as follows:

$$\mathbf{J}(f_{[i,j]}) = \begin{bmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_m(\mathbf{x})}{\partial x_n} \end{bmatrix} \tag{39}$$

, where $f_{[i,j]} = \mathbf{F}_{[i,j]}$ or $f_{[i,j]} = \mathbf{H}_{[i,j]}$. In many simulations, the Jacobian is computed through the actual derivative of the motion and transformation equations used in the simulation. The definition of derivative as the limit of the following equation

$$f'(x) = \frac{f(x+\Delta x) - f(x)}{\Delta x} \tag{40}$$

as Δx goes to zero allows for a numerical approximation of the Jacobians by replacing a small number for Δx (such as 1×10^{-4}).

3.3 Evaluation

Unlike, its linear counterpart, the Extended Kalman Filter, is not an optimal nonlinear filter. The EKF is vulnerable to divergence due to the Taylor Series approximations of its linearization. The EKF tends to diverge in instances where there is high initial uncertainty in the estimate. This is not to discount this nonlinear approximation, for in most cases the EKF provides a sufficient approximation.

4.0 The Particle Filter

4.1 Theory

4.1.1 Exact Bayesian Recursion

The solution of the filtering problem is to estimate the state of a system given all measurements up to time k , or in other words to construct the posterior p.d.f $p(x_k \mid Z_k)$. Exact recursive Bayesian estimation provides a solution in two recursive steps, prediction and update.

Prediction:

$$p(x_k \mid Z_{k-1}) = \int p(x_k \mid x_{k-1}) p(x_{k-1} \mid Z_{k-1}) dx_{k-1}$$

Update:

$$p(x_k \mid Z_k) = \frac{p(z_k \mid x_k) p(x_k \mid Z_{k-1})}{p(z_k \mid Z_{k-1})}$$

4.1.2 Sequential Monte Carlo Methods

Particle filters perform recursive Bayesian estimation directly on a set of samples that approximate the posterior density, $p(x_k \mid Z_k)$. These techniques, known as Sequential Monte Carlo methods, differ from exact recursive Bayesian estimation, in that they do not require the exact analytic solutions of distributions. An approximation of the posterior, achieved through the weighted sum of the Dirac Delta function, is sufficient through which to calculate interesting statistics, including the minimum mean squared error estimate (MMSE) of the state.

Monte Carlo Integration & Importance Sampling (M.C. Integration)	**Particle Filtering & Sequential Monte Carlo Methods (S.M.C)**
M.C. Integration provides a method for evaluating a multidimensional integral by sampling from a probability density $\pi(x)$. $I = \int g(\mathrm{X})d\mathrm{X} \ , \ \mathrm{X} \in \Re^n$ $= \int f(x)\pi(x)dx$	An estimate of the state of a system can be attained from: $E[f(x_k) \mid Z_k] = \int f(x_k)p(x_k \mid Z_k)dx_k$
While, it would be preferable to sample directly from $\pi(x)$, this is not always possible. When sampling directly is not an option, a Monte Carlo estimate is still feasible given that an importance density that is similar to $\pi(x)$ can be easily sampled and appropriately weighted. $I = \int f(x)\dfrac{\pi(x)}{q(x)}q(x)dx$ The integral approximation achieved by taking N independent samples from $q(x)$ becomes: $I_N = \sum_{i=1}^{N} f(x^i)w(x^i)$ $w(x^i)$ are the normalized importance weights: $\tilde{w}(x^i) \overset{\Delta}{=} \dfrac{\pi(x^i)}{q(x^i)}$ $w(x^i) \overset{\Delta}{=} \dfrac{\tilde{w}(x^i)}{\sum_{j=1}^{N}\tilde{w}(x^j)}$	In all but the more restrictive cases, it is not possible to sample directly from the posterior density, as a conceptual solution for the propagation of this density does not have a closed form solution or maybe be computationally expensive to generate [17]. Therefore, importance sampling is applied and an importance or proposal density is chosen: $E[f(x_k) \mid Z_k)] = \int f(x_k)\dfrac{p(x_k \mid Z_k)}{q(x_k \mid Z_k)}q(x_k \mid Z_k)dx_k$ Sampling from the proposal density, the Sequential Monte Carlo estimate is: $E[f(x_k) \mid Z_k)] = \sum_{i=1}^{N} f(x_k^i)w(x_k^i)$ $w(x_k^i)$ are the normalized weights: $\tilde{w}(x_k^i) \overset{\Delta}{=} \dfrac{p(x_k^i \mid Z_k)}{q(x_k^i \mid Z_k)}$ $w(x_k^i) \overset{\Delta}{=} \dfrac{\tilde{w}(x_k^i)}{\sum_{j=1}^{N}\tilde{w}(x_k^j)}$

4.1.2.1 Computation of Importance Weights

The importance weights are defined as a ratio of the posterior and the importance density. In order to construct an implementation of the particle filter one must simplify this ratio into quantities that are known and can be numerically computed.

(I) Defined by Monte Carlo Integration:

$$w(x_k^i) = \frac{p(x_k^i \mid Z_k)}{q(x_k^i \mid Z_k)}$$

(II) Simplify the joint posterior density:

$$p(X_k \mid Z_k)$$

a. Applying Bayes' Theorem:

$$p(X_k \mid Z_k) = \frac{p(Z_k \mid X_k)p(X_k)}{p(Z_k)}$$

b. Employ Markov Chain Property:

$$p(X_k | Z_k) = \frac{p(z_k, Z_{k-1})p(X_k)}{p(z_k, Z_{k-1})}$$

$$= \frac{p(z_k \mid X_k, Z_{k-1})p(X_k \mid Z_{k-1})}{p(z_k \mid Z_{k-1})}$$

$$= \frac{p(z_k \mid X_k, Z_{k-1})p(x_k, X_{k-1} \mid Z_{k-1})}{p(z_k \mid Z_{k-1})}$$

$$= \frac{p(z_k \mid X_k, Z_{k-1})p(x_k \mid X_{k-1})p(X_{k-1} \mid Z_{k-1})}{p(z_k \mid Z_{k-1})}$$

c. The states are propagated through a Markov process: $p(x_k \mid X_{k-1}) = p(x_k \mid x_{k-1})$
The observations are independent given the states: $p(z_k \mid X_k, Z_{k-1}) = p(z_k \mid x_k)$
The joint posterior reduces to:

$$p(X_k \mid Z_k) = \frac{p(z_k \mid x_k)p(x_k \mid x_{k-1})p(X_{k-1} \mid Z_{k-1})}{p(z_k \mid Z_{k-1})}$$

(III) Combining (I) and (II):

$$w(x_k^i) \qquad = \frac{p(z_k \mid x_k^i)p(x_k^i \mid x_{k-1}^i)p(X_{k-1}^i \mid Z_{k-1})}{q(x_k^i \mid Z_k)p(z_k \mid Z_{k-1})}$$

(IV) Let the importance density factorize such that

$$q(x_k^i \mid Z_k) \qquad = q(x_k^i \mid x_{k-1}^i, Z_k)q(x_{k-1}^i \mid Z_{k-1})$$

Then the weight equation becomes:

$$w(x_k^i) = \frac{p(z_k \mid x_k^i)p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{k-1}^i, Z_k)} * \frac{p(x_{k-1}^i \mid Z_{k-1})}{q(x_{k-1}^i \mid Z_{k-1})} * \frac{1}{p(z_k \mid Z_{k-1})}$$

(V) Finally, it is clear that:

$$\boxed{w(x_k^i) \quad \alpha \quad w(x_{k-1}^i)\frac{p(z_k \mid x_k^i)p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{k-1}^i, z_k)}}$$

The mathematical formulation above defines all parameters in terms of probability densities.

- Likelihood $\rightarrow$ $p(z_k \mid x_k)$
- Transition Density $\rightarrow p(x_k \mid x_{k-1})$
- Importance Density $\rightarrow q(x_k \mid x_{k-1}, z_k)$

A look at what these densities mean in terms of known statistics will reveal that these densities can be represented as integrals and are defined by the system and observation models, as well as known noise statistics.

4.1.2.1.1 Likelihood

The likelihood characterizes how well the available measurement z_k corresponds to the given prediction x_k. "The conditional PDF of z_k given x_k, $p(z_k \mid x_k)$, is defined by the measurement model and the known statistics of observation noise, v_k" [10].

$$p(z_k \mid x_k) = \int \delta(z_k - h_k(x_k, v_k))p(v_k)dv_k$$

The equation $h_k(x_k, v_k)$ is known as the observation equation. This function, which is a known model, relates the measurements to the state vector.

Derivation of Likelihood Equation

(I) Integration allows the likelihood to be written as:

$$p(z_k \mid x_k) = \int p(z_k, v_k \mid x_k) dv_k$$

(II) Applying Bayes' Theorem:

$$p(z_k \mid x_k) = \int \frac{p(z_k, v_k, x_k)}{p(x_k)} dv_k$$

(III) Implementing the Markov Chain Rule produces the following manipulation:

$$p(z_k \mid x_k) = \int p(z_k \mid v_k, x_k) p(v_k \mid x_k) dv_k$$

(IV) Assume v_k is independent of x_k, that is the observation noise is independent from past and current states, then $p(v_k \mid x_k) = p(v_k)$ and the likelihood becomes equivalent to:

$$p(z_k \mid x_k) = \int p(z_k \mid v_k, x_k) p(v_k) dv_k$$

(V) Manipulating as to incorporate the Dirac Delta function:

Note that:

$$p(z_k \mid v_k, x_k) = \begin{cases} 1 & when \quad h_k(v_k, x_k) = z_k \\ 0 & when \quad h_k(v_k, x_k) \neq z_k \end{cases}$$

Hence, it follows that:

$$\lim_{\sigma_{v_k}{}^2 \to 0} \delta(z_k - h_k(v_k, x_k)) = p(z_k \mid v_k, x_k)$$

Applying the above notion to (IV) gives the following result:

$$\boxed{p(z_k \mid x_k) = \int \delta(z_k - h_k(x_k, v_k)) p(v_k) dv_k}$$

The computation of the likelihood is dependent upon the probability of observation noise, as indicated by the derived equation.

There may not generally be only a concern with utilizing information from a single sensor. The extension to a multiple sensor problem is not difficult. Under the assumption of N independent sensors, centralized measurement fusion in the PF is simply computed by:

$$\prod_{j=1}^{S_N} p(z_k^j \mid x_k)$$

4.1.2.1.2 Transition Prior

The transition prior describes the evolution of the state. Where the likelihood is defined by the observation equation and known measurement noise, the transition prior $p(x_k \mid x_{k-1})$ is defined by the known system equation $f_{k-1}(x_{k-1}, w_{k-1})$ and process noise w_k. A mathematical representation of the transition prior, which can be derived in the same manner as the likelihood, takes on the form:

$$\boxed{p(x_k \mid x_{k-1}) = \int \delta(x_k - f_{k-1}(x_{k-1}, w_{k-1}))p(w_{k-1})dw_{k-1}}$$

4.1.2.1.3 Importance Density

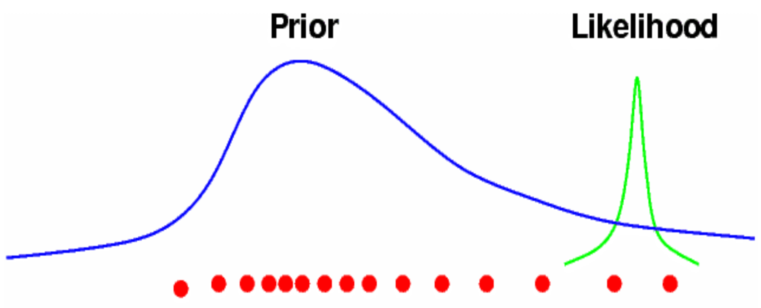
The proposal density $q(x_k \mid x_{k-1}, y_k)$ is a design choice involved in the implementation of a particle filter. In choosing a proposal density one would want a density as similar to the posterior as possible. However, the Monte Carlo Method only requires that the proposal and posterior have the same support. That is, particles that are associated with a probability greater than zero in the posterior should also be greater than zero in the proposal.

4.1.2.1.3.1 Transition Prior Proposal

The most commonly used proposal distribution is the transition prior $p(x_k \mid x_{k-1})$. Selection of the transition prior as the proposal density transforms the generic Sequential Importance Sampling algorithm into the algorithm known as the Bootstrap Filter. The prior is a popular choice, for it is easy to implement, causing the weight computation to reduce to a computation of the likelihood.

$$\boxed{w_k = w_{k-1}p(y_k \mid x_k)}$$

The decision to choose the prior as the sampling proposal disregards the fact that the proposal should take into account the most recent observation. As a result, the filter may potentially breakdown when the prior and likelihood do not overlap. Essentially, the likelihood weights improbable particles more highly than probable particles.



Above image is from http://cslu.cse.ogi.edu/publications/ps/UPF_CSLU_talk.pdf . This image illustrates the likelihood and prior disagreement that can lead to sample degeneracy.

4.1.2.1.3.2 Alternative Proposals

Alternatives to the transition prior proposal include using other fusion algorithms as the proposal density, including the EKF and Unscented Kalman Filter (UKF). These approaches may be favored, as they take into account the current measurement. However, using these proposals is both computationally more complex and as a result requires longer processing time.

4.1.2.2 Resampling

Resampling is the final step of an iteration of the particle filter. Without the resampling step, the algorithm so far derived is known as Sequential Importance Sampling (SIS). The resampling step becomes a necessity due to the fact that one must choose an importance density and not sample directly from the posterior. As a result of the inability to sample directly from the posterior, the variance of the weights increases over time, resulting in a phenomenon known as the Degeneracy Phenomenon. Essentially, this phenomenon causes the number of effective particles to decrease over time and may after many iterations result in only one particle with non-negligible weight. Hence, to prevent this degeneration resampling reselects particles with high importance weights. The following figure illustrates the resampling method for the reselection of particles with higher importance weights.

4.1.2.2.1 Resampling Schedules

(I) Deterministic resampling describes a schedule with fixed, interval times ($t_1, t_2, \ldots$).

(II) Dynamic scheduling is done when the effective number of particles is less than a predetermined threshold number.

4.1.2.2.1 Resampling Schemes

4.1.2.2.1.1 Systematic Resampling

Systematic Resampling is the computationally easiest resampling scheme. This ease comes from the fact that it uses the simple uniform distribution. The following equation is implemented when performing systematic resampling:

1) $U^i = \frac{i-1}{n} + U$, where U is a single random drawn from ((0,1/n]) and $i = 1, \ldots, n$.

The U^i is the i^{th} uniform sample. That is, the above equation randomly describes n uniform samples on the interval (0,1]. This sample is created by first drawing a random number on the interval (0,1/n]. The remaining uniform samples are then generated by adding $\frac{i-1}{n}$ to the randomly sampled U. Now, re-index the j^{th} particles based on the n i^{th} uniforms.

Even though the systematic resampling scheme is most popular due to the easy computation of a uniform distribution, other methods of resampling have been developed. The main differences between these resampling schemes are the distributions used to re-index the sample set.

4.1.2.2.1.2 Multinomial Resampling

In multinomial resampling, the duplication count of the j^{th} particle is determined by a multinomial distribution. That is, $N_1, \ldots, N_m$ are distributed according to the multinomial (n; $w_1, \ldots, w_m$).

Process:

1) Draw n independent uniforms $\{U^i\}$, $1 \le i \le n$ on the interval (0,1].
2) Set $I^i = D_w^{inv}(U^i)$ and $\xi^{i'} = \xi^{I^i} = \xi^{D_w^{inv}(U^i)}$, where D_w^{inv} is the inverse of the cumulative distribution function associated with normalized weight, that is, $D_w^{inv}(u) = i$ for $u \in (\sum_{j=1}^{i-1} w^j, \sum_{j=1}^{i} w^j]$.

4.1.2.2.1.3 Residual Resampling (Remainder Resampling)

1) Residual resampling is determined by the following equation:

$$N^j = \lfloor nw^j \rfloor + \overline{N^j}$$

N^j : total number of times the j^{th} particle is duplicated

$\overline{N^j}$: generated according to a multinomial distribution $Mult(n-R; \overline{w^1}, \dots, \overline{w^n})$

R : the sum of the integer part of the product of the total number of particles and the weight of the

j^{th} particle ($\sum_{j=1}^{n} \lfloor nw^j \rfloor$)

$\overline{w^j}$: $\dfrac{nw^j - \lfloor nw^j \rfloor}{n-R}, i = 1,\dots,n$

$\lfloor nw^j \rfloor$: integer part of the product of the total number of particles and the weight of the j^{th} particle.

4.2 Generic Particle Filter (Pseudo code)

- Initialization:
  - Draw N particles from a known initial distribution $p(x_1)$

- Update:
  - Upon receiving a measurement, evaluate the importance weights according to:

$$w_1^i = \frac{p(y_1 \mid x_1^i)p(x_1^i)}{q(x_1^i \mid y_1)}$$

  - Normalize importance weights:

$$\overline{w}_1^i = \frac{\dfrac{p(y_1 \mid x_1^i)p(x_1^i)}{q(x_1^i \mid y_1)}}{\sum_{j=1}^{N} \dfrac{p(y_1 \mid x_1^j)p(x_1^j)}{q(x_1^j \mid y_1)}}$$

  - Calculate Number of Effective Particles

  - If the number of effective particles falls below a threshold (IF $N_{eff} \langle N_{thr}$)
    Resample using a chosen Algorithm

- Prediction:
  FOR $i = 1 : N$

- Draw $x_k^i \sim q(x_k^i \mid x_{k-1}^i, y_k)$ by passing particles obtained after resampling through proposal density (Note: In the bootstrap filter, one would pass the particles through the system equation to draw these x_k^i particles.)

- Update:
  - Upon receiving a measurement, evaluate the importance weights according to:

$$w_k^i = \frac{p(y_k \mid x_k^i) p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{k-1}^i, y_k)}$$

  - Normalize importance weights:

$$\overline{w}_k^i = \frac{\dfrac{p(y_k \mid x_k^i) p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{k-1}^i, y_k)}}{\sum_{j=1}^{N} \dfrac{p(y_k \mid x_k^j) p(x_k^j \mid x_{k-1}^j)}{q(x_k^j \mid x_{k-1}^j, y_k)}}$$

  - Calculate the number of effective particles
  - If the number of effective particles falls below a threshold (IF $N_{eff} \langle N_{thr}$)

    Resample using a chosen Algorithm

5.0 Experiments

5.1 Maneuvering Targets

The algorithms described thus far are implementations using a single dynamic model. For tracking applications, a single model algorithm may not fully capture the true motion of a target. While, targets may move with a constant velocity for a period of time, a target may at a different time execute a turn, or in other words maneuver. In order to account for the varied motion, algorithms that implement multiple modes that are modeled as a Jump Markov System (JMS) are used.

5.2 System Models

The motion of the target is described at a given time instance by one of three dynamic models. The three models of target motion used are: (1) Constant Velocity Model (CV) (2) Clockwise Coordinated Turn Model (CT) (3) Counterclockwise CT Model. These three models are used in a switching manner to simulate a maneuvering target, where a regime variable $r_k \in \{1, 2, 3 \mid 1 = CV, 2 = Clockwise\, CT, 3 = Counterclockwise\, CT\}$

is used to account for the current mode at discrete time index k. The switching between models follows transitions given by a Markov chain with transition probabilities $\pi_{ij} = \Pr\{r_{k+1} = j \mid r_k = i\}, (i, j \in \{1, 2, 3\})$, such that $\pi_{ij} \geq 0, \sum_j \pi_{ij} = 1$. The evolution of the states x_k can then be described as

$$\mathrm{x}_k = f^{(r_k)}(x_{k-1}) + Gw_k$$

where the state vector x_k is defined as $\mathrm{x}_k = [x \quad y \quad \dot{x} \quad \dot{y}]$, w_k is the Gaussian process noise with covariance matrix $Q = q * I$, where I is an identity matrix and q is a scalar , and

$$G = \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix}$$

with T defined as the sampling interval. The discrete time instances where the state evolves according to the CV model with regime $r_k = 1$ can be gotten by replacing $f^{(r_k)}(x_{k-1})$ with the transition matrix:

$$F^{(1)}(\mathrm{x}_k) = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For evolution of the state with a constant turn rate the transition matrices are

$$F^{(j)}(\mathrm{x}_k) = \begin{bmatrix} 1 & 0 & \frac{\sin(O_k^{(j)} * T)}{O_k^{(j)}} & \frac{-(1-\cos(O_k^{(j)} * T))}{O_k^{(j)}} \\ 0 & 1 & \frac{(1-\cos(O_k^{(j)} * T)}{O_k^{(j)}} & \frac{\sin(O_k^{(j)} * T)}{O_k^{(j)}} \\ 0 & 0 & \cos(O_k^{(j)} * T) & -\sin(O_k^{(j)} * T) \\ 0 & 0 & \sin(O_k^{(j)} * T) & \cos(O_k^{j} * T) \end{bmatrix}, j = 2,3$$

with clockwise and counterclockwise turn rates given by

$$O_k^{(2)} = \frac{a_m}{\sqrt{\dot{x}^2 + \dot{y}^2}}$$

$$O_k^{(3)} = -\frac{a_m}{\sqrt{\dot{x}^2 + \dot{y}^2}}$$

where a_m is a constant maneuver acceleration parameter.

5.2.1 Interacting Multiple Model EKF (IMM-EKF)

The multiple dynamic models are implemented in the IMM-EKF as a process of interaction, filtering, and combination of outputs. The interaction begins each iteration of the algorithm by mixing the outputs from the previous step based upon the previous mode probabilities π_{ij}. Next, the inputs, derived from a single EKF with associated dynamic model, are filtered, resulting in both updated model probabilities and new inputs for the next step. Finally, the current model probabilities are used to compute a weighted average of the outputs from the individual filters, resulting in a combination and an estimate for discrete time step k.

5.2.2 Multiple Model PF (MM-PF)

The alterations in the MM-PF compared to the original algorithm are not drastic and do not significantly increase the computational complexity of the regular SIR algorithm. The added computations involve the determination of the regime variable for each of the n particles at discrete time step k. The mode is computed based upon: if $r_{k-1}^n = i$ and $u_n \sim U[0,1]$, then $\{m \in \{1,2,3\} \mid r_k^n = m\}$ such that: $\sum_{j=1}^{m-1} \pi_{ij} < u_n \leq \sum_{j=1}^{m} \pi_{ij}$. The particles are then propagated through the filter and predictions are determined that take into account the regime variable selected.

5.3 Measurement Models

The implemented simulation employs the ability to select sensors of four types that provide the following types of measurements: 1) Bearings Only 2) Range Only 3) Range and Bearings 4) Range, Elevation, and Azimuth. The measurement model described as

$$z_k^j = h^{(i)}(x_k) + v_k$$

where v_k is the measurement noise, can employ any of j sensors with $i = 1, \ldots, 4$ measurement models. The measurements models for the sensors can take any of the following forms:

$$h^{(1)}(x_k) = \left[\tan^{-1}\left(\frac{y_k - y^{s_j}}{x_k - x^{s_j}} \right) \right]$$

$$h^{(2)}(x_k) = \left[\sqrt{(y_k - y^{s_j})^2 + (x_k - x^{s_j})^2} \right]$$

$$h^{(3)}(x_k) = \left[\begin{array}{c} \tan^{-1}\left(\frac{y_k - y^{s_j}}{x_k - x^{s_j}} \right) \\ \sqrt{(y_k - y^{s_j})^2 + (x_k - x^{s_j})^2} \end{array} \right]$$

$$h^{(4)}(x_k) = \left[\begin{array}{c} \tan^{-1}\left(\frac{y_k - y^{s_j}}{x_k - x^{s_j}} \right) \\ \tan^{-1}\left(\frac{z_k - z^{s_j}}{\sqrt{(y_k - y^{s_j})^2 + (x_k - x^{s_j})^2}} \right) \\ \sqrt{(y_k - y^{s_j})^2 + (x_k - x^{s_j})^2 + (z_k - z^{s_j})^2} \end{array} \right]$$

where $(x^{s_j}, y^{s_j}, z^{s_j})$ is the position of sensor j.

5.4 Measures of Performance

Evaluations of the PF and EKF algorithms were primarily based upon average mean squared error (MSE) and variance (V) over separate Monte Carlo Runs. The mean squared error and variance computations are:

$$MSE_k = \frac{\sum_{k=1}^{\#TimeSteps} \sum_{j=1}^{\#Runs} \sqrt{(x_{k_{true}} - x^j_{k_{est}})^2}}{(\#\,Runs)(\#\,TimeSteps)}$$

$$V_k = \frac{\sum_{k=1}^{\#TimeSteps} \sum_{j=1}^{\#Runs} \sqrt{(x_{k_{true}} - x^j_{k_{est}})^2} - \sqrt{(x_{k_{true}} - x_{k_{avg}})^2}}{(\#Runs)(\#TimeSteps)}$$

5.5 Simulations

Comparison of the EKF vs. PF depends on several factors including: sensor position, “q”, and number of particles used in the particle filter.

Sensor position can degrade overall tracking performance for both EKF and PF. Overall tracking performance refers to RMSE and variance. A high tracking performance is described as having low RMSE and low variance. Performance is degraded when the sensors are not lined up normal to the target track. When the sensors are lined up in parallel with the direction of motion of the target, performance is minimized.

In this simulation, overall tracking performance was at a minimum when: 1) the two sensors were positioned side by side on all four sides of the target’s track; 2) the two sensors were positioned on opposite sides of the target track parallel with the target’s motion. Alternatively, tracking performance was maximized when either: 1) one or more of the sensors were positioned close to the target track so at no time during the simulation was the sensor in parallel with the target’s motion; 2) the two sensors were positioned at right angles on the outside of the target tack ensuring at least one sensor was normal to the target at all times.

In the two cases mentioned previously where performance is minimized, the EKF outperformed the PF in nearly every situation. Despite the EKF’s performance over the PF in this sensor setup, it is important to remember the RMSE and the variance were still very large.

When the sensors are positioned so that performance is maximized the results weren’t as one-sided toward the EKF as when the sensors were positioned poorly. A noticeable trend emerged from this sensor setup. The PF outperformed the EKF whenever “q” was set to 1.6×10^{-6} no matter how many particles were used (50, 100, 500).

When the PF was set to 50 particles in this performance maximization sensor setup, the EKF showed better performance for the “q” values of 0.5, 1, and 5. As the number of particles increased, the performance of the PF increased (with the exception that simulation time increased dramatically as a result from more computational complexity). When the number of particles was increased to 100 the EKF outperformed or did just as well as the PF with the exception of when “q” was set to 1.6×10^{-6} as mentioned above. Once the number of particles was set to 500 the PF either outperformed or did just as well as the EKF.

Examples of Poor Sensor Alignment

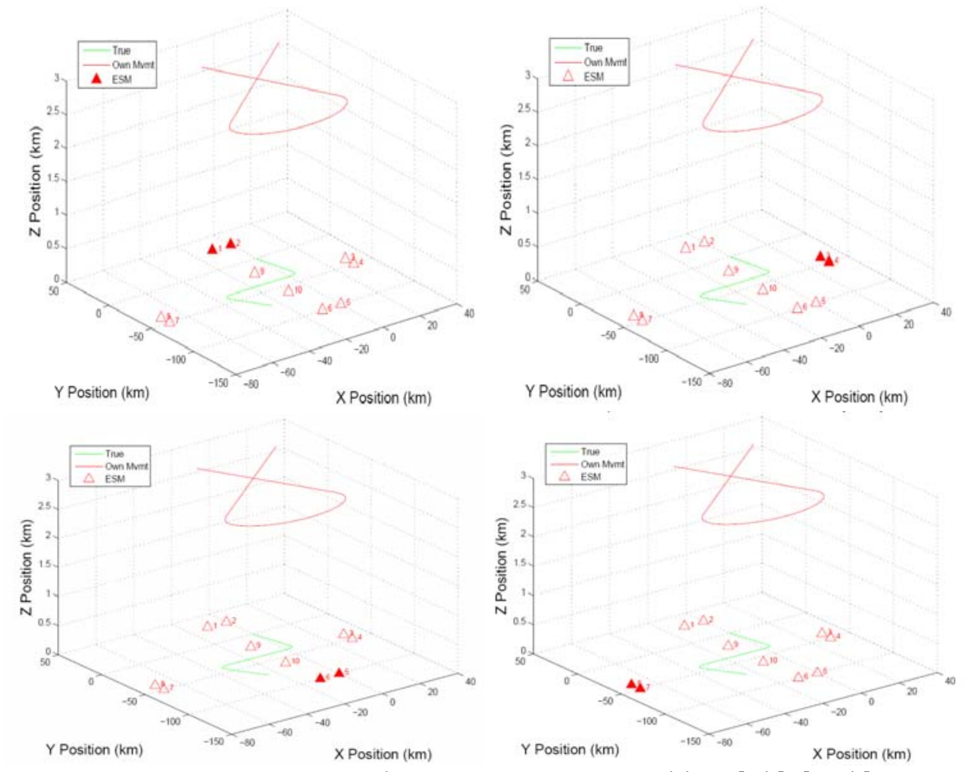


Figure 1. Two sensors positioned side by side.

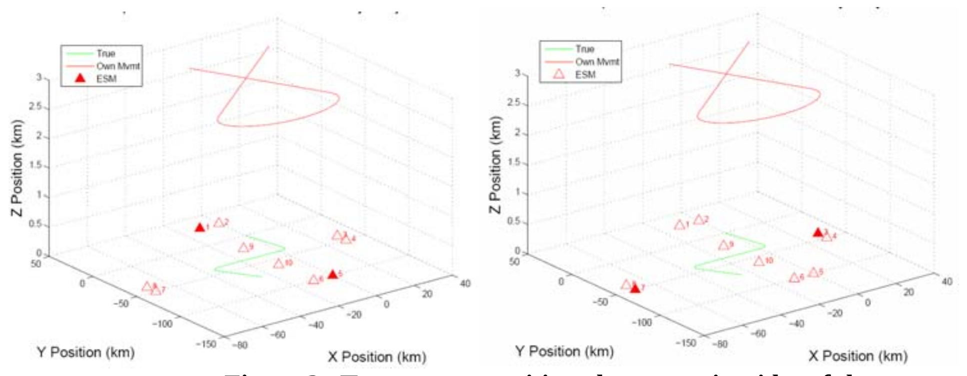


Figure 2. Two sensors positioned on opposite sides of the target track.

Examples of Proper Sensor Alignment

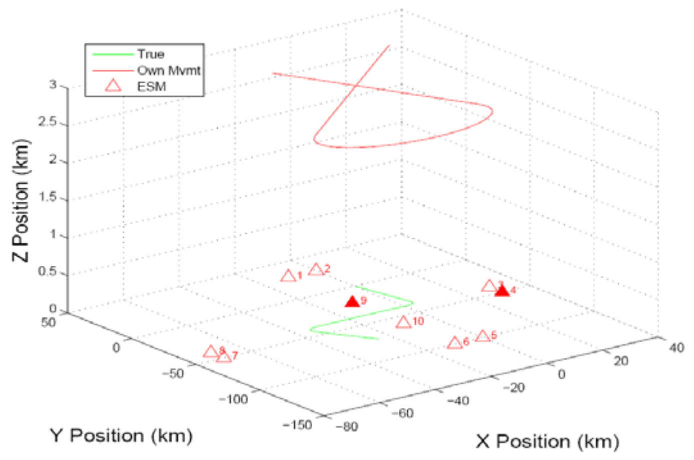


Figure 3. At least one sensor positioned close to the track path.

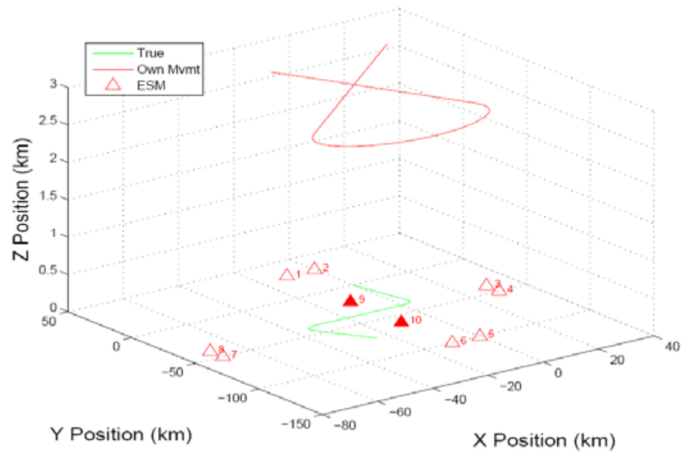


Figure 4. Both sensors positioned close to the track.

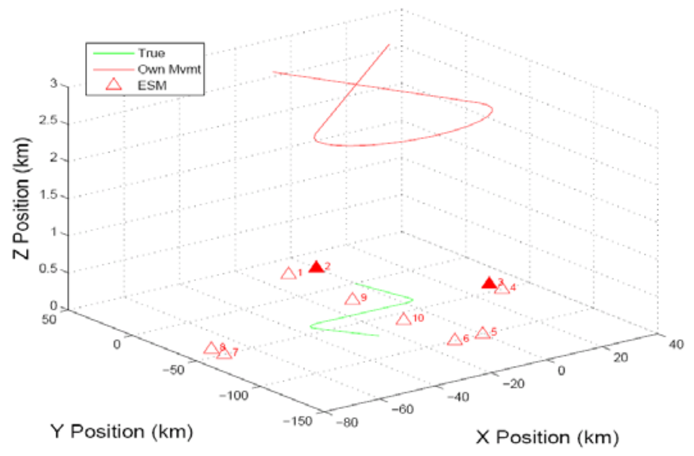


Figure 5. Sensors positioned at right angles centered on the target track.

Typical Results with Poor Sensor Alignment

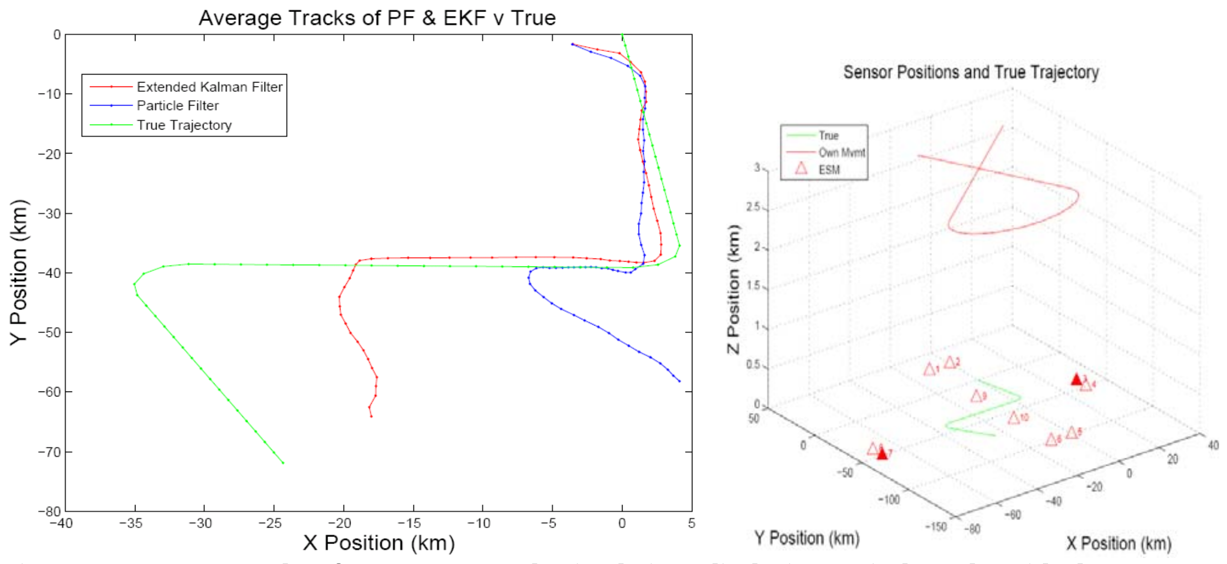


Figure 6. Average tracks of 100 Monte Carlo simulations displaying typical results with the two sensors positioned on opposite sides of the target track.

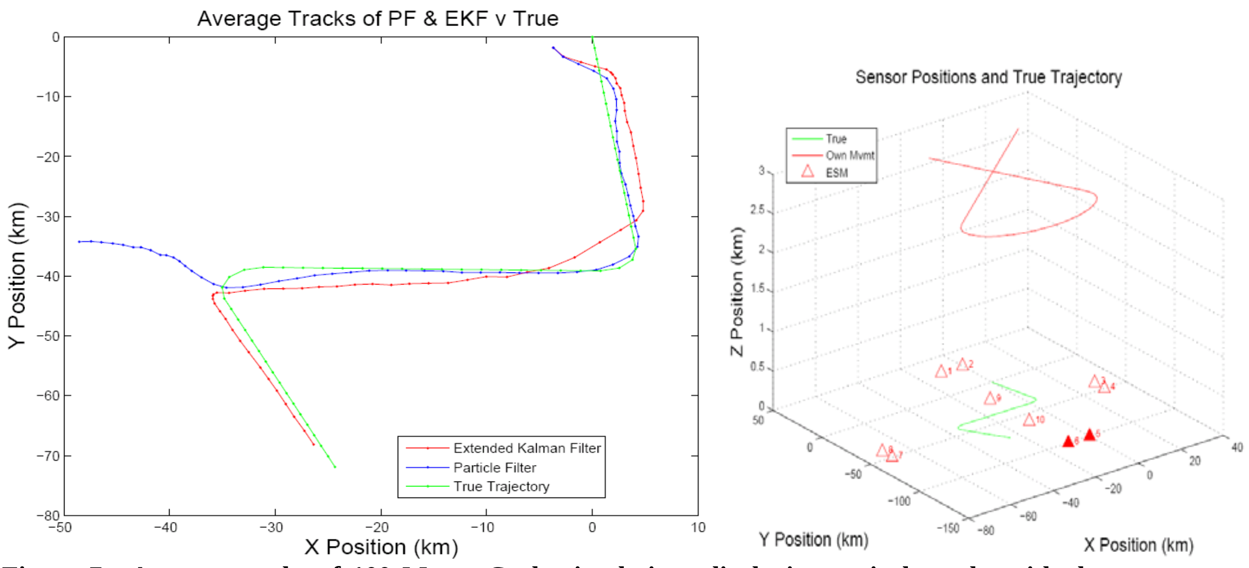


Figure 7. Average tracks of 100 Monte Carlo simulations displaying typical results with the two sensors positioned side by side.

Effects of “q” and the Number of Particles with Properly Aligned Sensors

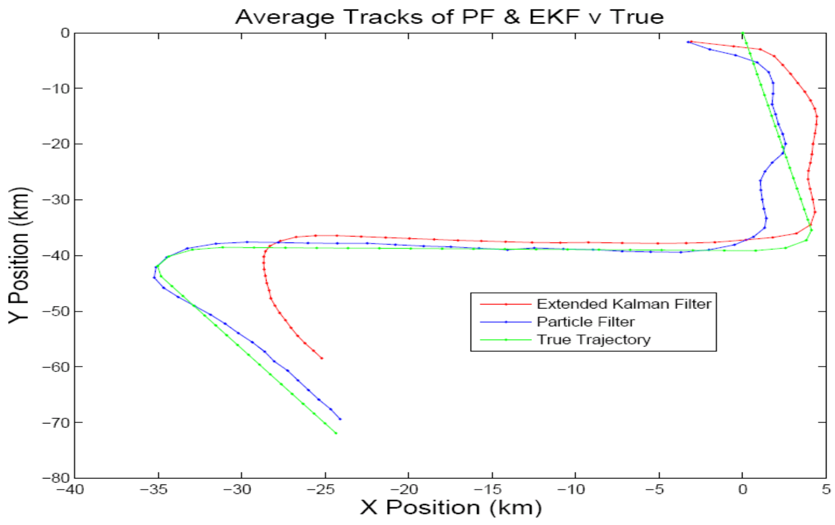


Figure 8. Average tracks of 100 Monte Carlo simulations with q = 1.6×10^{-6} and the number of particles set to 50.

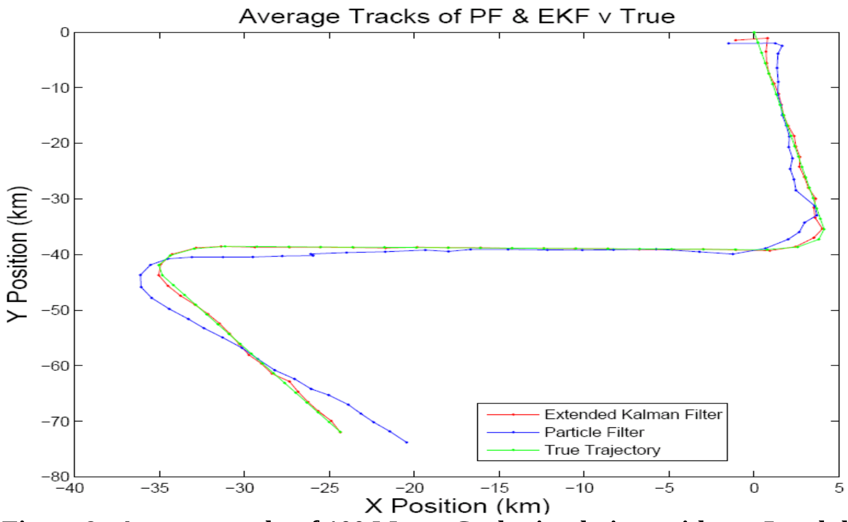


Figure 9. Average tracks of 100 Monte Carlo simulations with q = 5 and the number of particles set to 50.

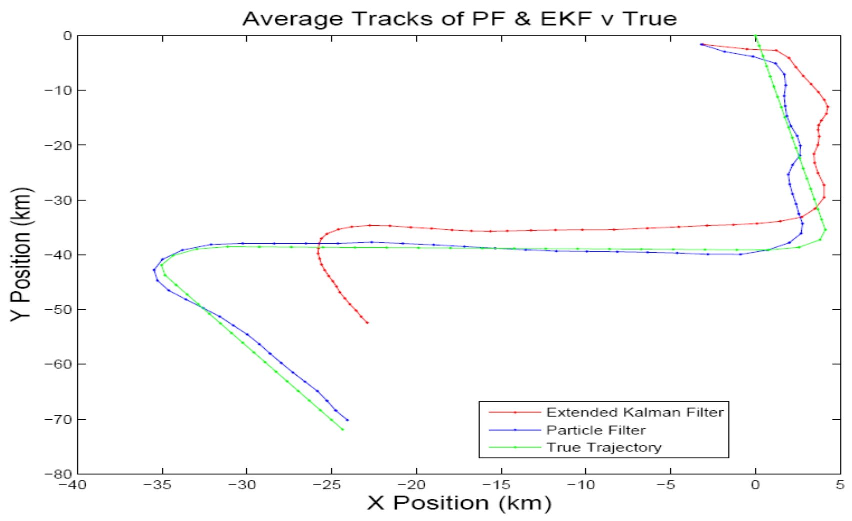


Figure 10. Average tracks of 100 Monte Carlo simulations with q = 1.6×10^{-6} and the number of particles set to 100.

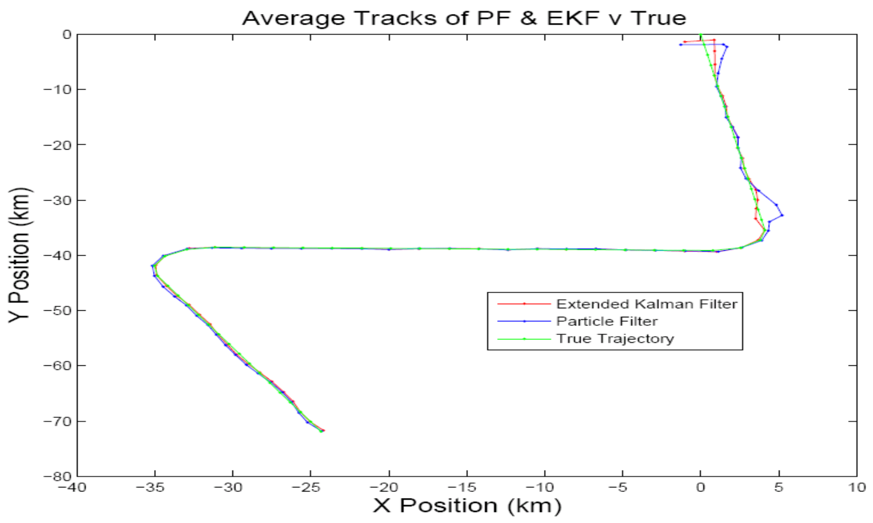


Figure 11. Average tracks of 100 Monte Carlo simulations with q = 5 and the number of particles set to 100.

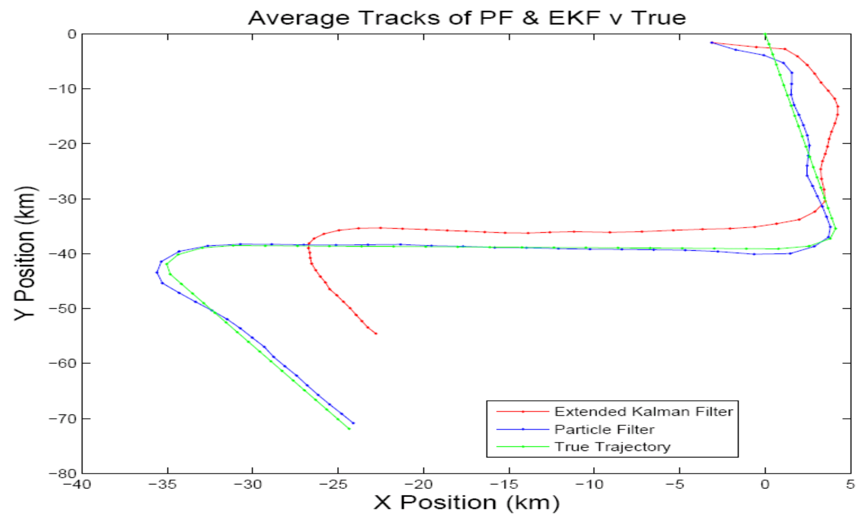


Figure 12. Average tracks of 100 Monte Carlo simulations with q = 1.6×10^{-6} and the number of particles set to 500.

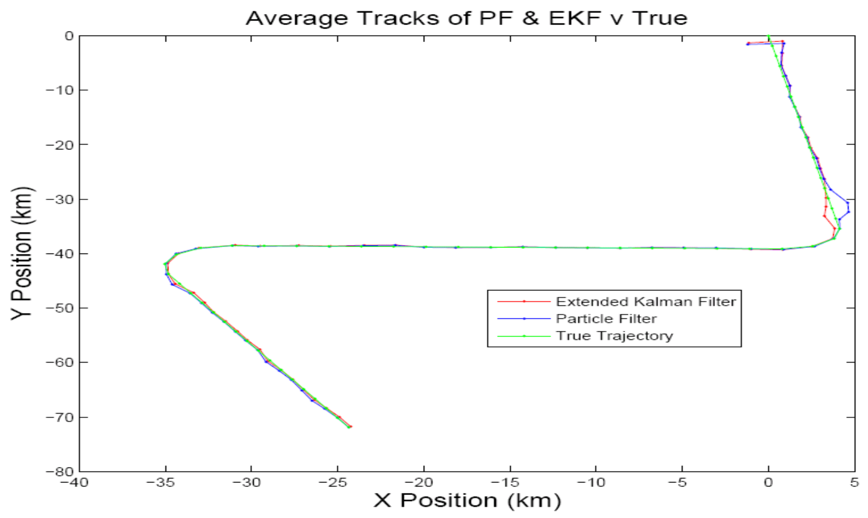


Figure 13. Average tracks of 100 Monte Carlo simulations with q = 5 and the number of particles set to 500.

6.0 Chief Scientist Briefing Results

On August 30, 2007, a briefing was given by Mark Alford to the Chief Scientist of AFRL/IF, Dr. John S. Bay. The following is a summary of that briefing.

The overall in-house program is being referred to as "Fusion Techniques and Non-Linear Filtering (FTNLF)." This overall program encompasses past and future in-house efforts along with component contractual efforts. The outline of the briefing consisted of going over the purpose, particle filtering, and progress, followed by a discussion of the major in-house efforts and then the component efforts, followed by conclusions and future directions.

The primary purpose of the FTLNF In-House Program is to investigate and develop promising innovative technologies that hold promise for improvements in Air Force target tracking and multi-sensor fusion capabilities. Current permanent staffing is shown in the following organizational chart.

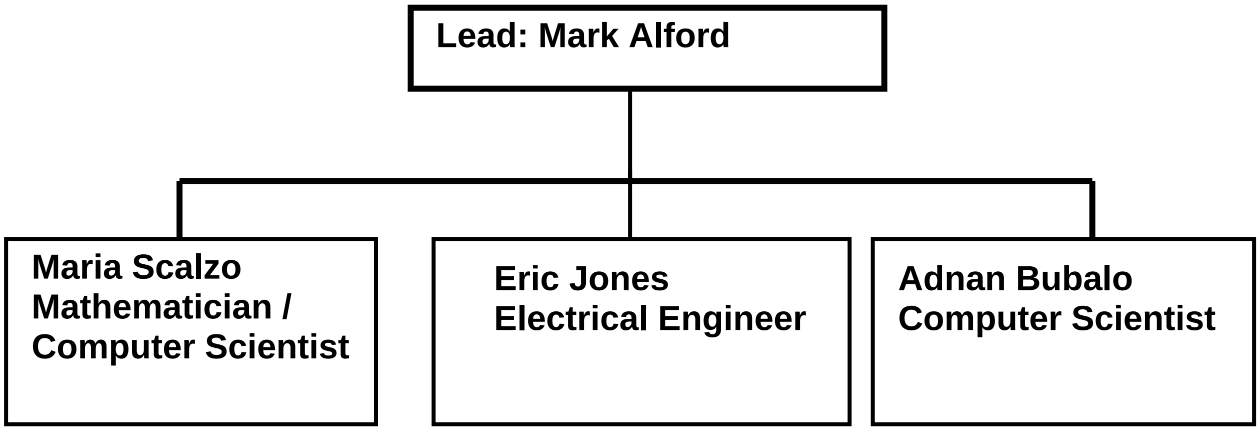

Lead: Mark Alford
Maria Scalzo Mathematician / Computer Scientist
Eric Jones Electrical Engineer
Adnan Bubalo Computer Scientist

Figure 14. In-House FTLNF program staffing

Also, there are typically two summer students hired each year to help support this program. In the summer of 2007, Christopher Poore and Becky Bailey worked on nonlinear filtering techniques, and their reports are included in the Appendices of this report. There is also University/Contractual support through Dr. Pramod Varshney, Syracuse University, Black River Systems Company, and Numerica Corporation.

Figure 15 shows an overview of the fusion scenario.

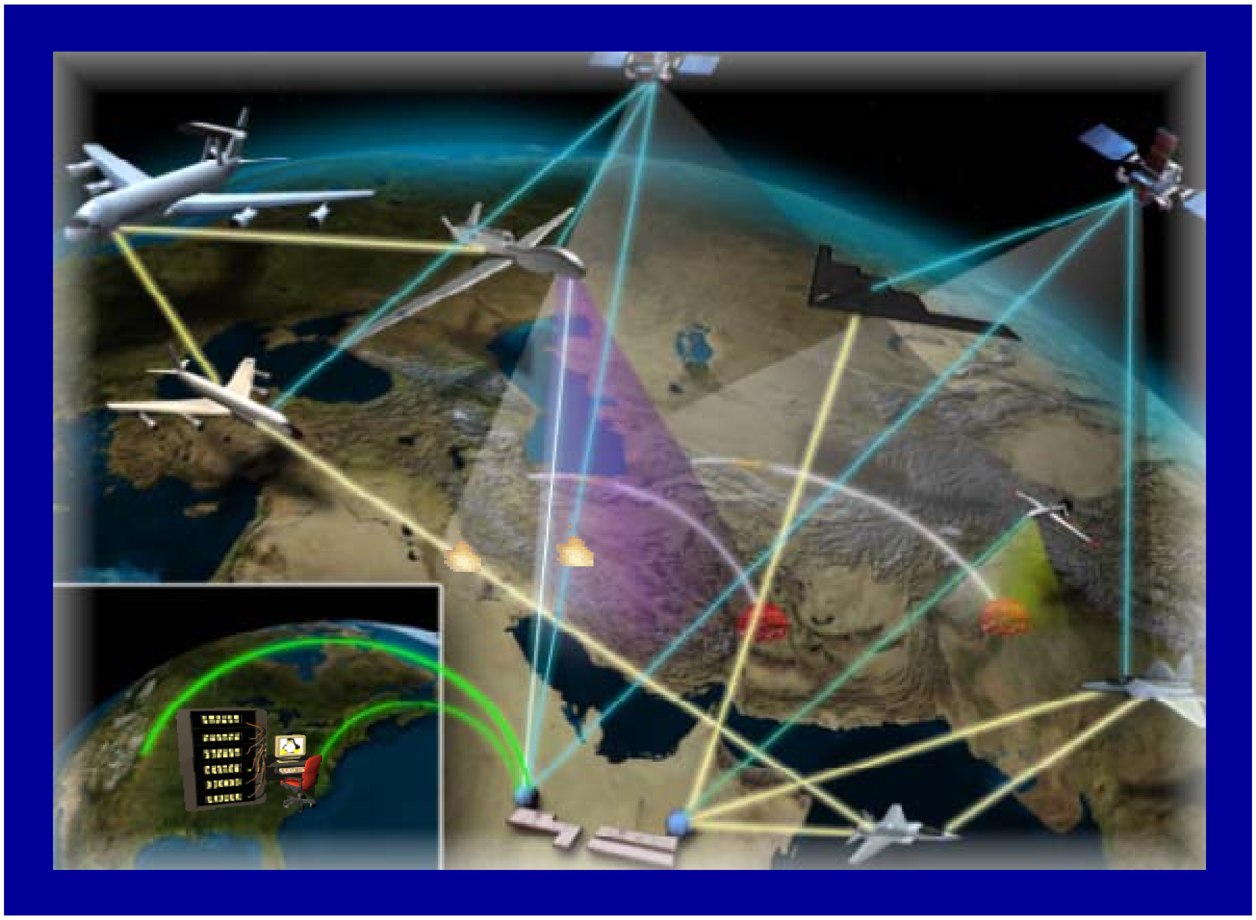


Figure 15. Overview of the Fusion Scenario

Multiple assets are shown conducting surveillance over the ground, air, space and sea. Some have overlapping coverage, some do not. This shows the idea of sharing information between platforms and with ground and space based systems. These platforms and systems are monitoring events in both war and peace time environments. Examples of events are the fires on the ground. Terrain features are also important in this scenario. Both centralized and decentralized fusion centers are being employed.

The problem statement is that Nonlinear Non-Gaussian Processes (NNGP) present a major challenge in all types of military problems. This is because the real-world is nonlinear and non-Gaussian, despite assumptions made in most conventional fusion algorithms. The FTLNF In-House program is addressing this problem by researching and developing tracking filters that do not presume a linear Gaussian world. Perhaps the most famous of these is the particle filter. The theory behind particle filtering has been around for a long time, namely in the form of Monte Carlo Markov Chain based approaches, however, it resurfaced with the name particle filtering in 1993 with Neil Gordon's thesis referenced in [10].

Figure 16 depicts an urban clutter picture which shows how nonlinear things can be. Imagine a target moving through this environment. Clearly this is a nonlinear environment. With a little imagination, you can almost tell that it is non-Gaussian as well. Consider a camera, radar, infrared sensor, unmanned aerial vehicle (UAV), or other sensing device overlooking this scenario. The question is, how do we discern critical situations in this sort of environment. This is exactly where it is expected that new nonlinear filtering techniques will help.

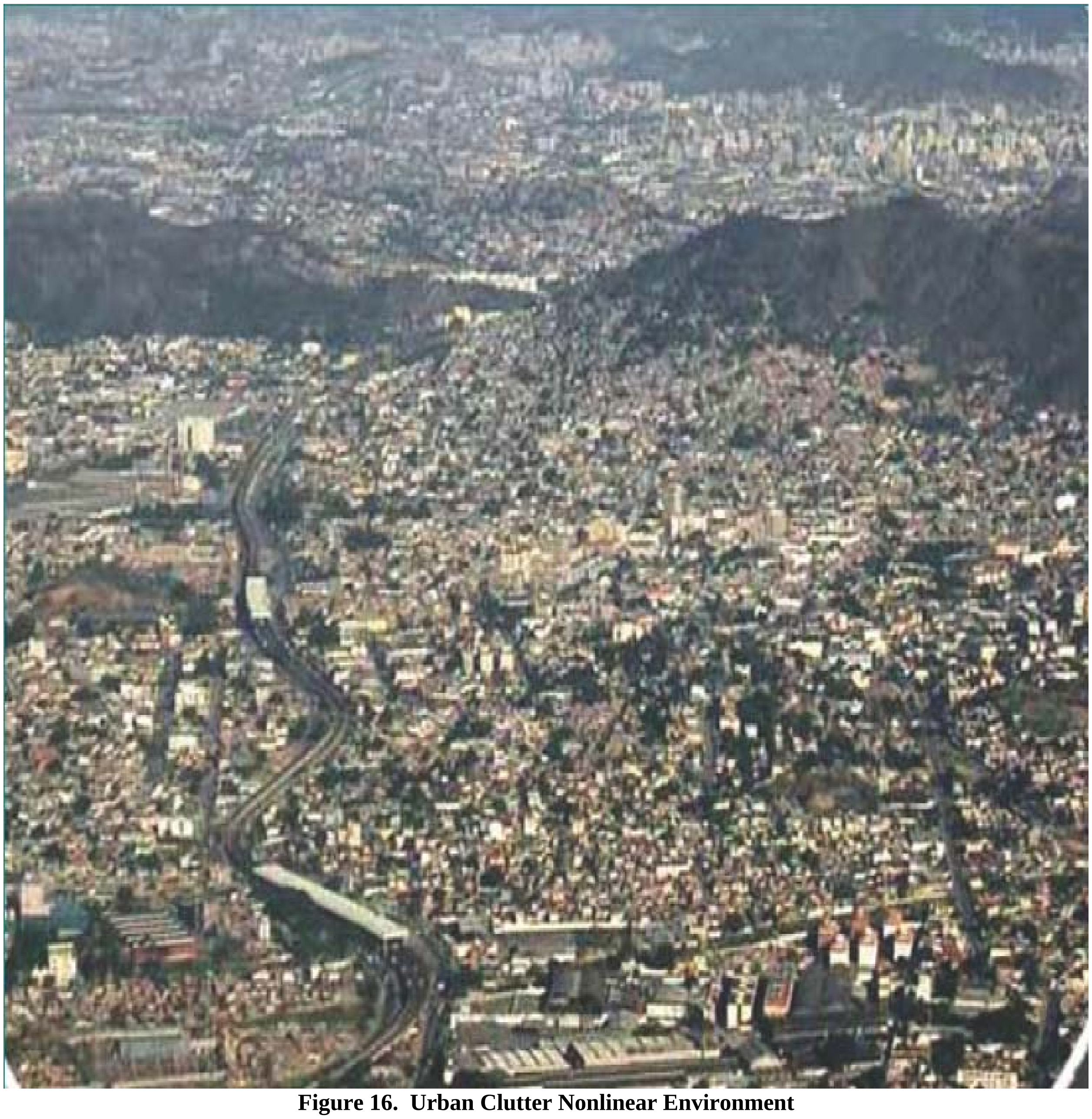


Figure 16. Urban Clutter Nonlinear Environment

The particle filter is the focus of the FTNLF research. Particle Filtering holds great promise to provide enhanced capability for non-linear, non-Gaussian tracking conditions. Considerable work is required to verify the situations where the Particle Filter will provide improved performance. Developing tools and techniques to investigate Particle Filter performance relative to other filtering techniques is a major emphasis of the effort under the FTNLF In-House program. Other non-linear filters are also being investigated for comparison purposes to determine the conditions under which they represent a better choice than the particle filter.

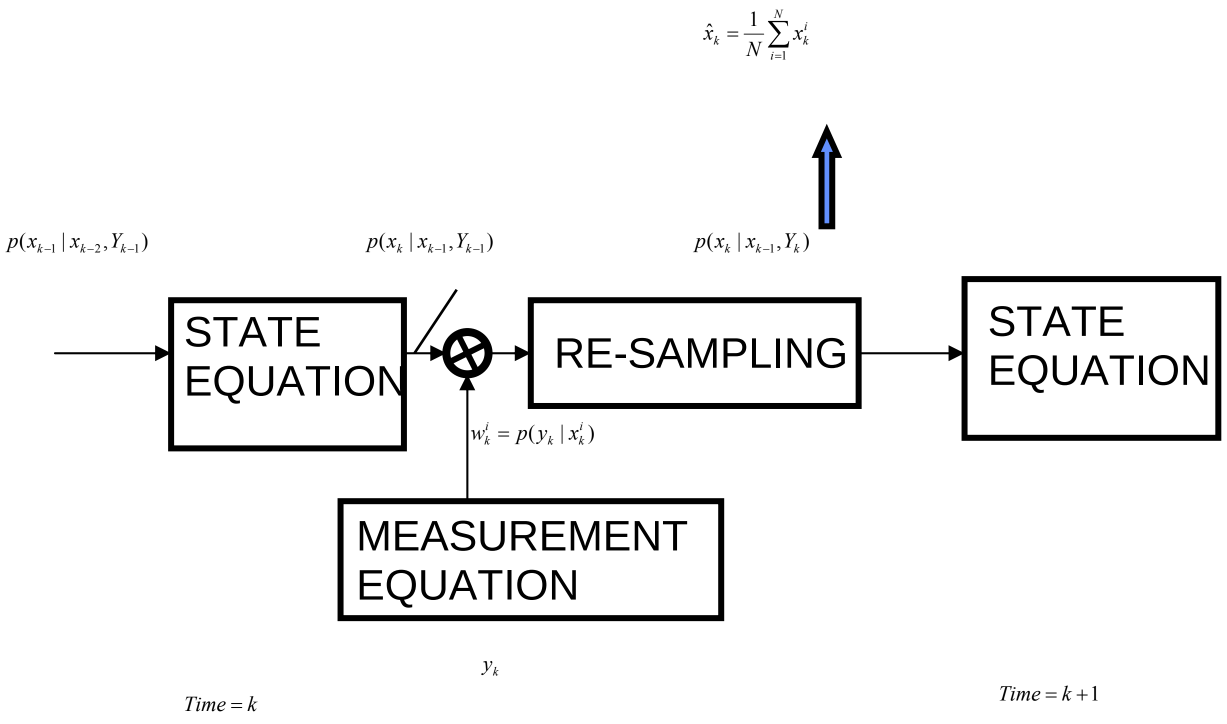
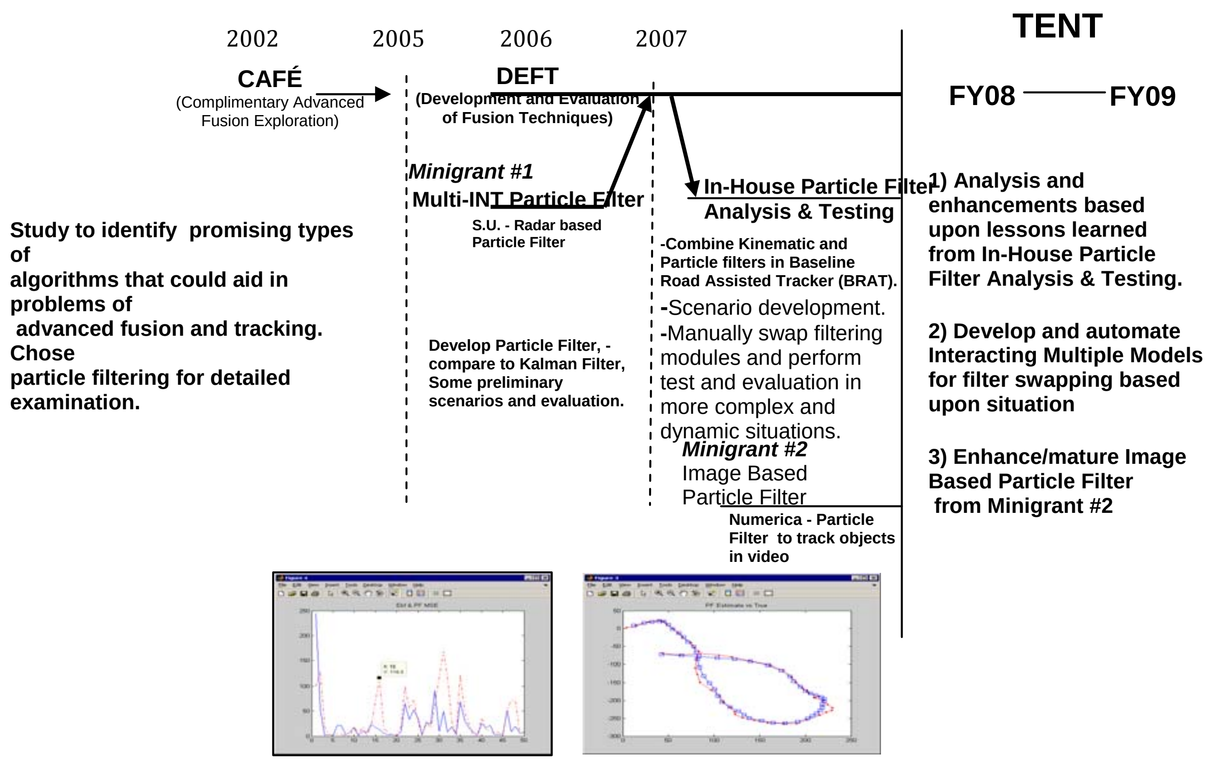


Figure 17. Particle Filter: A Block Diagram

Figure 17 shows a block diagram of the particle filter. The input is the probability density function (pdf) of the target state x consisting of position, velocity, acceleration, jerk and other attributes. The pdf goes through the state update equation to propagate the initial state (prediction step) from previous time step k-1 up to time k. A measurement comes in consisting of, for example, range-bearing or bearings only from a radar, infrared, or Electronic Support Measures (ESM) sensor. This measurement is passed through the measurement equation to form a likelihood that measurement y was generated by state x. The weighted, updated pdf is then resampled. When this measurement updated density is formed, the mean is calculated to create the track. The whole process is then repeated for the next time step and measurement update.

The FTNLF In-House program progress in Particle Filter investigation, development, and verification has focused on the development and utilization of a simulation capability including:

1. Development of a MATLAB simulation capability in coordination with Syracuse University
2. Preliminary investigation of Particle Filtering capabilities



TENT
2002
2005
2006
2007
CAFÉ
(Complimentary Advanced Fusion Exploration)
DEFT
(Development and Evaluation of Fusion Techniques)
FY08
FY09
Minigrant #1
Multi-INT Particle Filter
S.U. - Radar based Particle Filter
In-House Particle Filter Analysis & Testing
Study to identify promising types of algorithms that could aid in problems of advanced fusion and tracking. Chose particle filtering for detailed examination.
-Combine Kinematic and Particle filters in Baseline Road Assisted Tracker (BRAT).
-Scenario development.
-Manually swap filtering modules and perform test and evaluation in more complex and dynamic situations.
Develop Particle Filter, - compare to Kalman Filter, Some preliminary scenarios and evaluation.
1) Analysis and enhancements based upon lessons learned from In-House Particle Filter Analysis & Testing.
2) Develop and automate Interacting Multiple Models for filter swapping based upon situation
3) Enhance/mature Image Based Particle Filter from Minigrant #2
Minigrant #2
Image Based Particle Filter
Numerica - Particle Filter to track objects in video

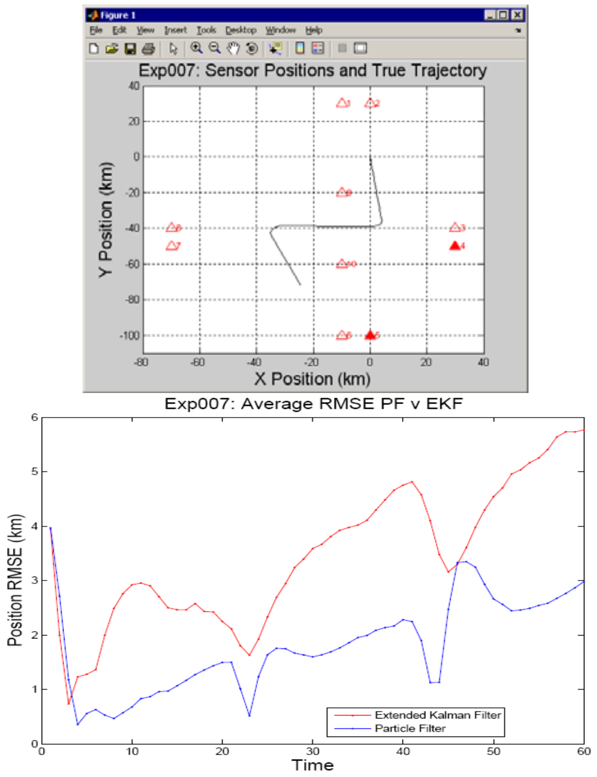
Figure 18. FTNLF In-House Program Timeline

Figure 18 shows a timeline of the in-house programs comprising FTNLF. The first program was Complementary Advanced Fusion Exploration (CAFÉ) which was a study to identify promising types of algorithms that could aid in problems of advanced fusion and tracking. It was found that Particle Filtering was an area that was not being worked within the Information Directorate at Rome, and was chosen as an area for detailed examination. A final report was published for CAFÉ and a small examination of particle filtering was done. It showed that the particle filter outperformed the extended Kalman filter for nonlinear trajectories and Rayleigh Noise. In fact, it was found that the Unscented Particle Filter (UPF) outperformed the Unscented Kalman Filter (UKF) and the Particle Filter.

The follow-on to CAFÉ was Development and Evaluation of Fusion Techniques (DEFT) whose objective was to understand the Particle Filter in more depth. The result was the development of a particle filter to compare to Extended Kalman filters with some preliminary scenarios and evaluation. The Multi-INT Particle Filter Minigrant was an outgrowth of DEFT. DEFT developed a two-dimensional particle filter for in-house testing and analysis. Other programs initiated under DEFT were the In-House Particle Filter Analysis and Testing (IHPFAT) fallout money program and another Minigrant to do Image Based Particle Filtering. IHPFAT has the objective of combining the kinematic and particle filter with the Baseline Road Assisted Tracker developed by Black River Systems Company. Additionally, Black River was tasked with helping AFRL with appropriate scenario development. Once these software tools are integrated, the idea was to manually swap filtering modules and perform test and evaluation in more complex and dynamic situations. The Image Based PF Minigrant is just getting under way, so not much progress has been made, but the idea is to use a particle filter to track objects in video.

Tracking Evasive Nonlinear Targets is a new two year In-House program to perform analysis and enhancements based on lessons learned under previous efforts. The ultimate goal is to develop and automate an Adaptive Nonlinear Tracking System (ANTS) for filter swapping/parallelization based upon situation. It is also a goal to enhance/mature Image Based Particle Filtering from the second Minigrant.

Under DEFT there were some accomplishments that lead to the conclusion that particle filtering has promise. DEFT developed a two-dimensional particle filter for in-house analysis and testing. Under DEFT, the team performed Monte Carlo simulation runs to test the performance of the Particle Filter (PF) as compared to the EKF for multiple bearings only sensors (ESM sensors). A comparison was made of tracking error variance based on Root Mean Square (RMS) position error. Sample results (where the PF outperformed the EKF) are shown in Figures 19 and 20.



```
                --------------
                Experiment 007
                --------------
Experiment number 7 in series of 15.
Number of particles: 50
Number of runs: 100
Number of time steps: 60
Active Sensors   Type    x pos    y pos  glint
probability
4                 1      30       -50     0.0
5                 1       0       -100    0.0
Number of active sensors:  2
Maneuverability variable q: 1.600e-005
PF lost track 0 times out of 100, or 0.00% of the
time.
EKF lost track 28 times out of 100, or 28.00% of the
time.
     AvgTime/Time_Step       Avg RMSE      Variance
PF       0.0417                1.795          3.000
EKF      0.0042                3.449         47.631
Processing took  9.91 times longer for PF than for
EKF.
```

Figure 19. DEFT Simulation Conditions: 2 ESM Sensors

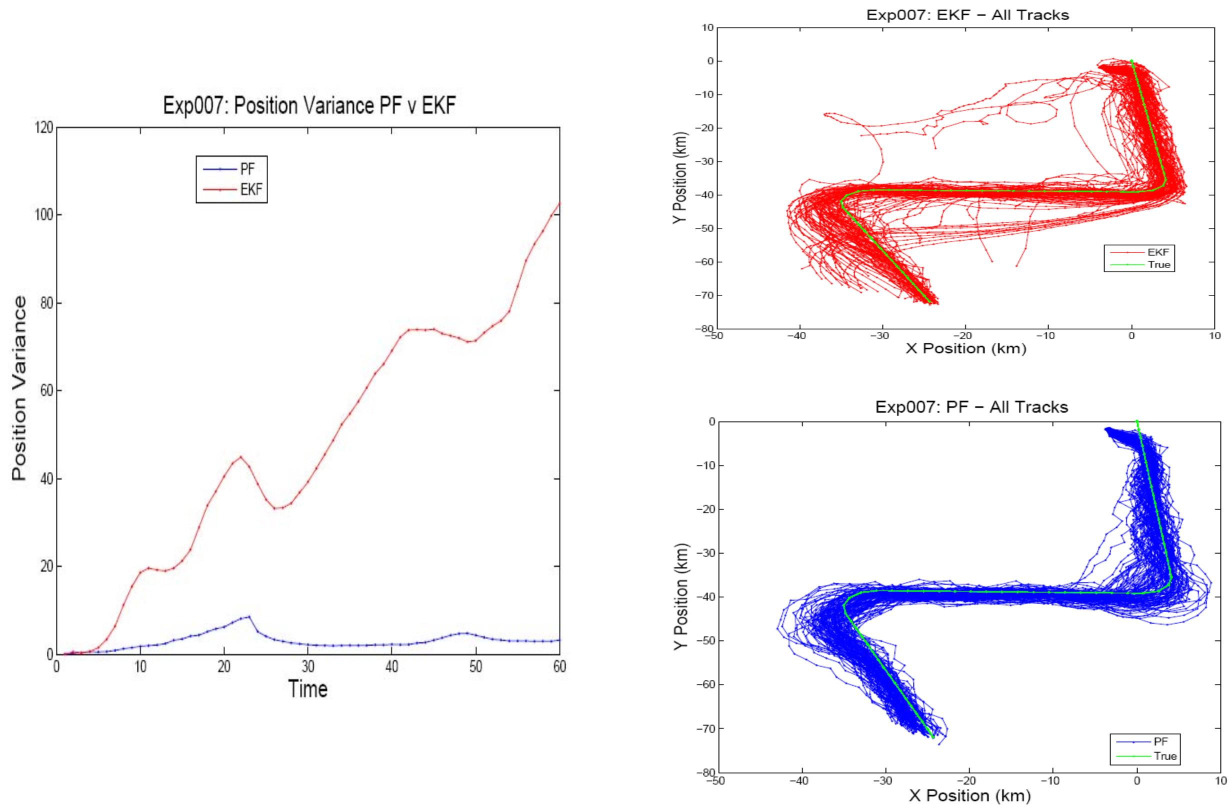
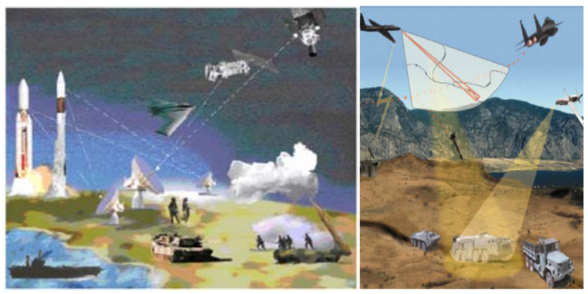


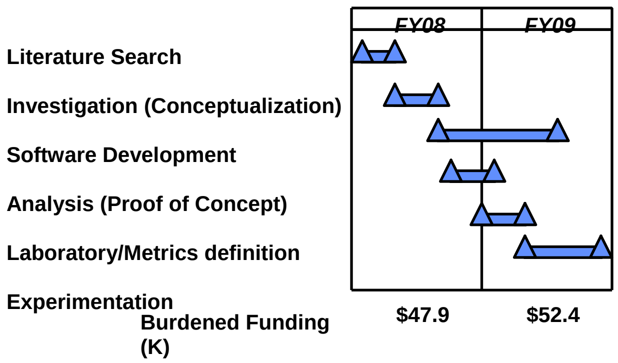
Figure 20. Results: variance is much less for PF than EKF

Figure 7 shows that the tracking error variance is much better for the PF than the EKF, requiring 10x as much processing time. The result is for 50 particles, 100 Monte Carlo runs, two bearings only sensors. The EKF lost track 28% of the time whereas the PF did not lose track at all. The variance and RMSE versus average computation time per time step are compared in Figure 6. Use of High Performance Computers (HPCs) is being worked with IFTC.

The follow on effort to DEFT is TENT. TENT will develop tools and techniques such as automated filter selection and improved tracking for nonlinear: target motion, platform motion, and sensor dynamics. TENT will investigate the applicability of image based tracking, enhance existing algorithms, and leverage contributing technologies such as Multi-INT Particle Filtering.

Tracking Evasive Nonlinear Targets (TENT)


Tracking Targets in Non-linear Motion



Description:
This in-house effort will develop tools and techniques to automate filter selection to improve tracking by reducing susceptibility to non-linearities in target motion, platform motion, and sensor dynamics. It will also investigate the applicability of image based particle filter tracking as well as enhancing existing algorithms.

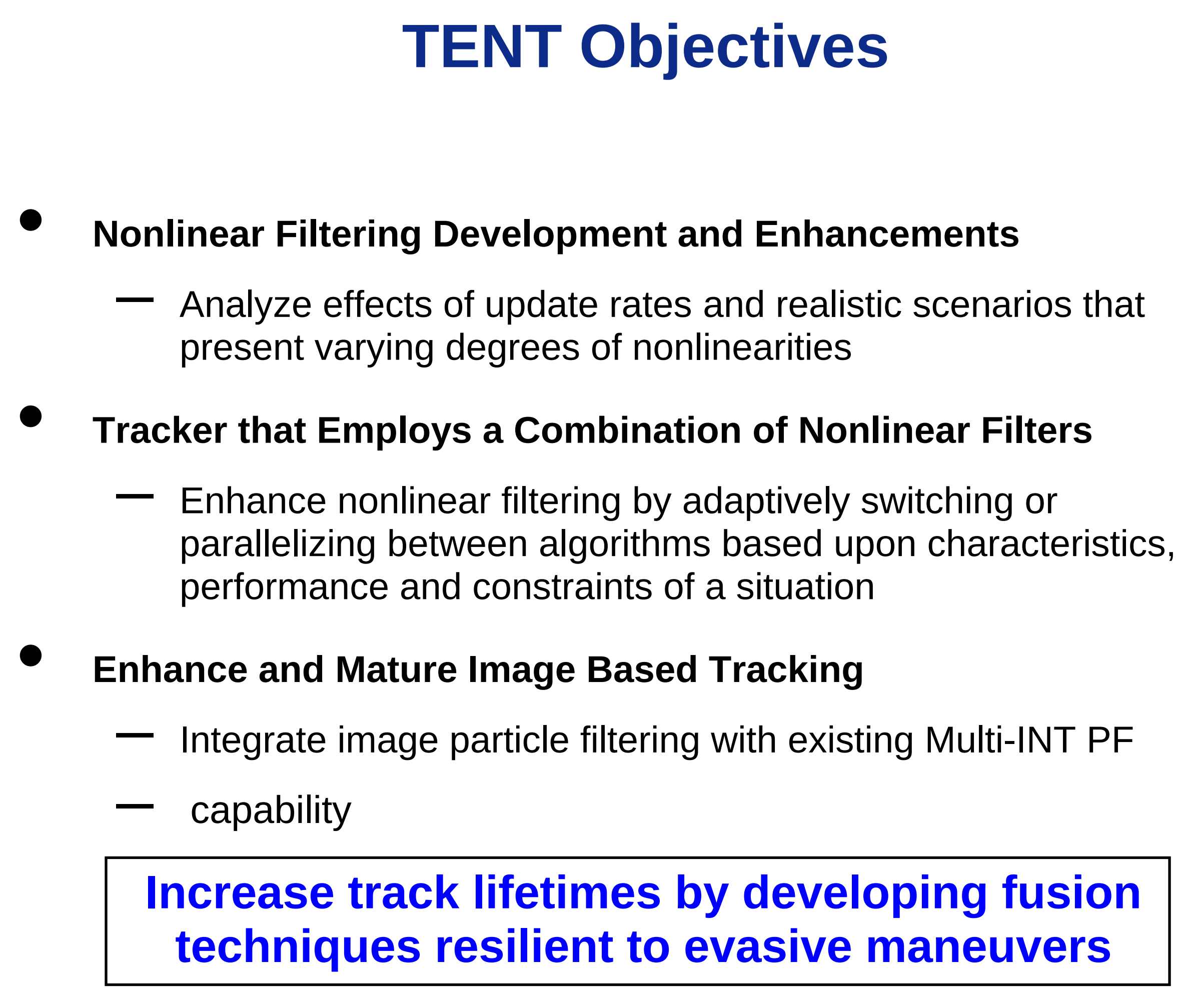
Technologies: Multi-INT Particle Filtering

Benefits to the War Fighter:

- Provides new capabilities for tracking targets moving in nonlinear, non-Gaussian manner
- Promotes longer tracks by having less target loss during maneuvers (fewer dropped tracks)
- Dismount Tracking

Figure 21. TENT Quad Chart

A quad chart for TENT is shown in Figure 21. The TENT objectives are Nonlinear Development and Enhancements, Tracker that employs a combination of nonlinear filters and enhance and mature image based tracking. The overarching theme is to increase track lifetimes by developing fusion techniques resilient to evasive maneuvers.



TENT Objectives

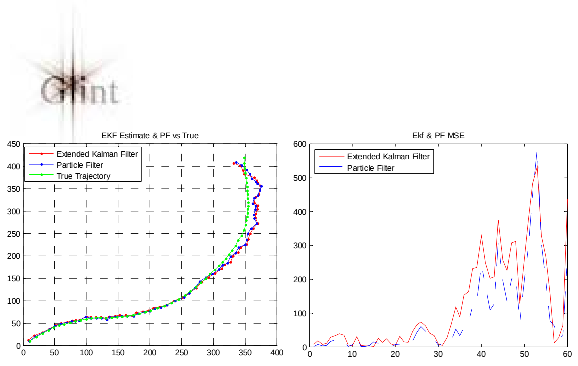
- **Nonlinear Filtering Development and Enhancements**
  - Analyze effects of update rates and realistic scenarios that present varying degrees of nonlinearities
- **Tracker that Employs a Combination of Nonlinear Filters**
  - Enhance nonlinear filtering by adaptively switching or parallelizing between algorithms based upon characteristics, performance and constraints of a situation
- **Enhance and Mature Image Based Tracking**
  - Integrate image particle filtering with existing Multi-INT PF
  - capability

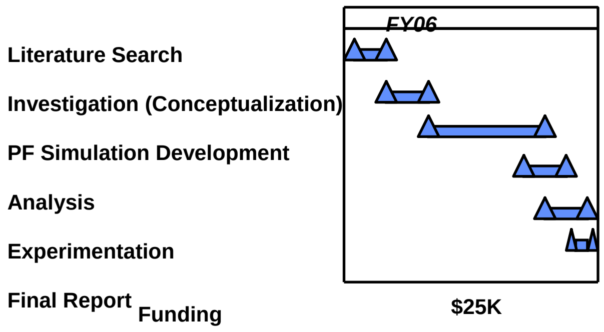
Increase track lifetimes by developing fusion techniques resilient to evasive maneuvers

Figure 22. TENT Objectives

The Multi-INT Particle Filtering Minigrant was performed in-house in conjunction with Syracuse University. The final report is included as an appendix to this report. A quad chart for that effort is shown in Figure 23.

Multi-INT Particle Filtering Minigrant





Description: This project compared EKF & PF in the presence of glint noise. Changes in the aspect toward the radar can cause irregular electromagnetic wave reflections, resulting in significant variation of radar reflections. This phenomenon gives rise to outliers in angle tracking, and it is referred to as target glint. We adopt a commonly used model for glint noise. This model consists of one Gaussian with high probability and small variance and another with small probability and very high variance.

Technologies: Multi-INT Particle Filtering

Benefits to the War Fighter:

- Nonlinear tracking in the presence of glint
- ELINT IMINT fusion using particle filtering
- Multiple ESM Sensor fusion
- Established In-House Particle Filtering simulation capability
- Multiple Heterogeneous sensor simulations
  - Range Bearing
  - Bearings Only
  - Range Only
- Established a baseline approach for testing particle filtering algorithms in-house

Figure 23. Minigrant Quad Chart

The purpose of the Multi-INT Particle Filtering Mini-Grant was to establish a means of fusing data from Multiple Intelligence sources using Particle Filtering as a basis. Traditional methods of data fusion involved extensive use of the Kalman Filter. The basic Kalman Filter made the assumptions that the underlying processes were linear and Gaussian. For example, Electronic Intelligence (ELINT) and Imagery Intelligence (IMINT) information require consideration of the probabilistic characteristics of the underlying sources of those types of information. It can safely be said that the underlying processes are nonlinear and non-Gaussian. Particle Filters were typically focused on the single information source tracking problem. No attempt had been made to combine the state estimation capabilities of Multi-INT information using Particle Filters. Investigation of the feasibility of using Particle Filtering as a basis for Multi-INT fusion was accomplished by developing a MATLAB simulation and experimenting with promising sensor configurations. Progress was made in a number of areas including:

1. Established an in-house software simulation capability to do particle filtering
2. Established a baseline approach for testing particle filtering algorithms in-house

Several conclusions have been reached as a result of FTNLF effort thus far. The problem of non-linear non-Gaussian behavior is a difficult problem that cannot be addressed through conventional methods. Investigation of Particle Filtering and other recent innovations in non-linear filtering has show promise. Considerable study is necessary to reach conclusive results based on algorithm performance. Current and future work is focusing on developing sufficient tools and understanding of the mathematical nature of non-linear filtering techniques to fully determine the conditions under which these techniques will lead to improved performance.

This work has also pointed to some future directions. One of the key areas that needs to be investigated is the application of Measures (or Degrees) of Nonlinearity (MoN, DoN) to determine when it make sense to use a specific type of nonlinear filter (e.g. PF). Also required is the investigation into specific types of scenarios that would dictate the use of new nonlinear filtering techniques. Once this is established, the next step is implementation of a wide variety of nonlinear filtering techniques within a software suite for further investigation and experimentation. Finally, extensive documentation of results and conclusions from using the experimental suite is required.

References

[1] Athalye, A., M. Bolic, S. Hong, P.M. Djuric. “Generic Hardware Architectures for Sampling and Resampling in Particle Filters.” Eurasip Journal on Applied Signal Processing. vol 17, pp 2888-2902, 2005.

[2] Bailey T. "Tim Bailey's Home Page [Internet]," Sydney (Australia); 19 July 2006. Available from: http://www.acfr.usyd.edu.au/homepages/academic/ tbailey/

[3] Bay, John S. *Fundamentals of Linear State Space Systems* Boston: WCB/McGraw-Hill, 1999

[4] Brandstadt, J.C. “Kalman Filter Notes” Black River Systems Company. 3 June 2006.

[5] Brookner E., *Tracking and Kalman Filtering Made Easy*, John Wiley & Sons, Inc., 605 Third Avenue, New York, NY, 1998.

[6] Cappe, O., Douc, R., Moulines, E. "Comparison of Resampling Schemes for Particle Filtering."[online] <http://www.cmap.polytechnique.fr/~douc/Page/Research/cdm.pdf >. June 16,2006

[7] Chow SM, Ferrer E, and Nesselroade, JR., "An Unscented Kalman Filter Approach to the Estimation of Nonlinear Dynamical Systems Models". Available from: http://psychology.ucdavis.edu/labs/ferrer/pubs/mbr_ inpress.pdf

[8] de Freitas, Nando, R. Van der Merwe. Upf_demos. [online] \\Lfs-projects\offices\IF_directorate\IFE_division\Div_Public\NewFolder (5)\upf_demos (Original)

[9] de Freitas, Nando. *Monte Carlo Methods II. [Online] <http://www.cs.ubc.ca/~nando/papers/mctalkII.pdf.> August 8, 2006.*

[10] Gordon, N.J., D.J. Salmond, and A.F.M Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," IEE Proc.-F, vol. 140, no. 2, pp. 107-113, 1993.

[11] Julier SJ and Uhlmann JK. "A New Extension of the Kalman Filter to Nonlinear Systems," OXI 3PJ, The University of Oxford, Oxford, UK. Available from: http://www.cs.unc.edu/~welch/kalman/media/pdf/Julier 1997_SPIE_KF.pdf

[12] "Kalman filter." Wikipedia, The Free Encyclopedia. 5 Jul 2006, 18:43 UTC. Wikimedia Foundation, Inc. 6 Jul 2006. Available from: http://en.wikipedia.org/w/index.php?title=Kalman_filter&oldid=62228105

[13] Kleinbauer R., "Kalman Filtering Implementation with Matlab," Helsinki (Finland); Universität Stuttgart; 2004 Nov. Available from: http://elib.uni-stuttgart.de/opus/volltexte/2005/2183/pdf/kleinbauer.pdf

[14] LaViola JJ Jr, "A Comparison of Unscented and Extended Kalman Filtering for Estimating Quaternion Motion," Brown University Technology Center for Advanced Scientific Computing and Visualization, Providence, RI, 02912.

[15] "Nonlinearity-Compensation Extended Kalman Filter and Its Application to Target Motion Analysis," OKI Technical Review, 1997; Vol 63, No. 159. Available from: http://www.oki.com/en/otr/html/nf/otr-159-06.html

[16] Pitt, M.K., N. Shephard. "Filtering via Simulation: Auxiliary Particle Filters." Journal of the American Statistical Association. vol. 94, no.446, pp.590-599, 1999.

[17] Ristic, Branko, Sanjeev Arulampalam, and Neil Gordon. Beyond the Kalman Filter: Particle Filters for Tracking Applications. Boston: Artech House,2004.

[18] Sankaranarayanan, A.C., R. Chellappa, & A. Srivastava. "Algorithmic and Architectural Design Methodology for Particle Filters in Hardware." [Online] <http://www.iccd-conference.org/proceedings/2005/045_srivastavaa_particle filters.pdf>. July 27, 2006.

[19] Simon D., "Kalman Filtering." Embedded Systems Design. 1 June 2001, 11:43 EDT. CMP. 14 June 2006. Available from: http://www.embedded.com/story/OEG20010529S0118

[20] Van der Merwe, Rudolph, Arnaud Doucet, Nando deFreitas, and Eric Wan. "The Unscented Particle Filter." Technical Report CUED/F-INFENG/TR380.16 Aug. 2000. Cambrdige University Engineering Department.10June 2005. <http://cslu.cse.ogi.edu/publications/ps/UPF_CSLU_talk.pdf> July 15, 2006.

[21] Vermaak J., "The Unscented Kalman Filter and its Mixture Version," 2005 January. Available from: www-sigproc.eng.cam.ac.uk/~jv211/papers/ukf_note_05.pdf

[22] Wan EA and van der Merwe R., *Kalman Filtering and Neural Networks*. "Chapter 7: The Unscented Kalman Filter,"; 1-49. Available from: http://citeseer.ist.psu.edu/cache/papers/cs/22073/http:zSzzSzcslu.cse.ogi.eduzSzpublicationszSzpszSzwan01a.pdf/wan01chapter.pdf

[23] Wan, EA and van der Merwe R. "The Unscented Kalman Filter for Nonlinear Estimation," Oregon Graduate Institute of Science & Technology; Beaverton, OR. Available from: http://cslu.cse.ogi.edu/nsel/ukf/test.html

[24] Welch G. and Bishop G. "An Introduction to the Kalman Filter," TR 95-041, Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC, 2006 July 24.

[25] "Woodbury Matrix Identity." Wikipedia, The Free Encyclopedia. 5 Jul 2006, 18:43 UTC. Wikipedia Foundation, Inc. 6 Jul 2006. Available from: http://en.wikipedia.org/wiki/Woodbury_matrix_identity

[26] Zhang Z., "Parameter Estimation Techniques: A Tutorial with Application to Conic Fitting," Image and Vision Computing Journal 1997; Vol 15, No. 1: 59-76. Available from: http://www-sop.inria.fr/robotvis/personnel/ zzhang/Publis/Tutorial-Estim/Main.html

[27] Zuo L., "An Introduction to Particle Filters and Some Applications," In: Cooperation between AFRL Rome Research Site and Syracuse University; 2006 Feb 14; Syracuse, NY.

Appendix I. Multi-INT Particle Filtering Minigrant report

Particle Filtering Experimentation

Summary

This project develops several data fusion and target tracking algorithms for a surveillance system that consists of multiple heterogeneous sensors. Algorithms based on the classical extended Kalman filter (EKF) and on the emerging non-Gaussian and nonlinear particle filtering (PF) techniques have been implemented. These algorithms have been tested in the practical case where a target maneuvers from time to time and an Interacting Multiple Model (IMM) framework is used. We have also tested them in the presence of glint noise. Changes in the aspect toward the radar can cause irregular electromagnetic wave reflections, resulting in significant variation of radar reflections. This phenomenon gives rise to outliers in angle tracking, and it is referred to as target glint. We adopt a commonly used model for glint noise, the Gaussian mixture model. This model consists of one Gaussian with high probability and small variance and another with small probability of occurrence and very high variance.

The performances of the EKF and the particle filter have been compared through extensive simulation experiments. The results show that for highly non-linear measurements, such as those from multiple bearing-only sensors, particle filter exhibits a superior data fusion and tracking performance than the EKF. However, if the system receives measurements from a radar (both bearing and range measurements), the EKF and the PF have very similar tracking accuracy, and the EKF is a more desirable choice, considering that it requires much less computation than the PF, and has a much easier real-time implementation.

Index Terms

Target tracking, Particle Filter, Extended Kalman Filter (EKF), Interacting Multiple Model (IMM), glint noise, Gaussian Mixture Model (GMM), sensor networks

I. INTRODUCTION

Combining the information from multiple heterogeneous sensors can lead to more accurate tracking results than using a single sensor. To fuse these heterogeneous and non-linear measurements, there are many tracking algorithms, of which the most commonly used is the classical method called the extended Kalman filter (EKF) [1], where the non-linear measurement model and/or nonlinear motion model are linearized via Taylor series expansion, and the noises are approximately assumed Gaussian. On the other hand, a Monte-Carlo simulation based recursive estimation algorithm, the particle filtering (PF) algorithm [2, 3] has emerged as a very promising technique to solve the non-linear and non-Gaussian filtering problem. It has been shown that using highly nonlinear measurements, such as bearing-only measurements, the PF outperforms the EKF [2].

Here in this project, we compare the tracking performance of the EKF and the PF under various situations, where different combinations of sensor measurements are available for data fusion. Different types of sensors are considered, including range-only sensors, bearing-only sensors, and radars that provide both range and bearing measurements. Besides the non-linearity in the measurements, non-Gaussian measurement noise, the glint noise modeled as a Gaussian mixture, has been used in the experiments. In addition to the relatively easy case where the target moves at nearly a constant velocity, we investigate the difficult case where targets maneuvers and an IMM algorithm has to be used. Through simulation experiments, we demonstrate that the particle filter has superior performance during the first several steps after initialization. In steady state, when the data are highly nonlinear bearing-only measurements, the PF still outperforms the EKF. However, whenever radar data are available, the PF has very similar steady-state performance as the EKF in terms of MSE.

II. SENSOR NETWORK SETUP

Since multiple heterogeneous sensors are connected to form a sensor network, it is very important to take advantage of the information from multiple sources. Here we adopt one of the most common data fusion schemes, namely the centralized fusion scheme. In a centralized fusion process, all the sensors transmit their raw measurements, such as range, and bearing to the fusion center, as shown in Fig. 1. After collecting all these measurements, the fusion center fuses them to form a new and more accurate estimate of the target state. The fusion is accomplished by the tracker in a very natural way. Namely all the raw measurements and their associated accuracies are used to update the target state. The tracker only needs to adjust its measurements equation to reflect that measurements are from multiple heterogeneous sensors. The centralized fusion scheme is optimal in the sense that no information is lost during the fusion process, since the unprocessed raw measurements are transmitted to the fusion center.

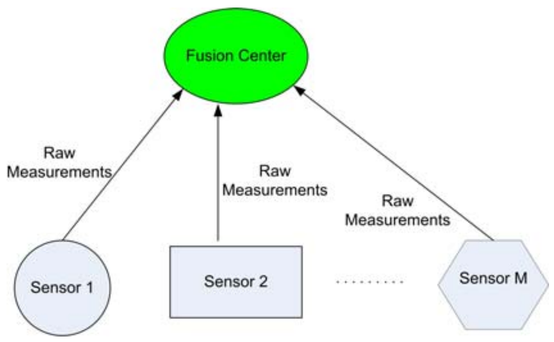


Fig. 1 Centralized fusion process.

III. System Models

A. Target Motion Model

The maneuvering target motion is modeled by three switching dynamics models whose evolution follows a Markov chain, also called a Jump Markov System (JMS) [4, 7]. We assume that at any time, the target moves according to one of s = 3 dynamic behavior models: (a) Constant Velocity (CV) motion model, (b) clockwise Coordinated Turn (CT) model, and (c) anticlockwise CT model. Let S= {1, 2, 3} denotes the set of three models for the dynamic motion. Then, the target dynamics can be written as

$$\mathbf{x}_k = f^{(r_k)}(\mathbf{x}_{k-1}) + \mathbf{v}_k$$

where $\mathbf{x}_k$ is the state vector defined by $\mathbf{x}_k = [x \;\; \dot{x} \;\; y \;\; \dot{y}]$, k denotes the discrete time index, and $r_k \in S$ is the regime variable taking effect in the time interval (k−1, k], with transition probabilities $\pi_{ij} \overset{\Delta}{=} \Pr\{r_{k+1} = j \mid r_k = i\}, (i, j \in S)$, such that $\pi_{ij} \geq 0, \sum_{\mathrm{j}} \pi_{ij} = 1$. The initial probabilities are denoted by $\pi_i \overset{\Delta}{=} \Pr\{r_0 = i\}$ for $i \in S$, and $\pi_i \geq 0$, $\sum_i \pi_i = 1$. v_k denotes the white Gaussian noise with covariance matrix Q:

$$Q = q\begin{bmatrix} 1/3T^3 & 1/2T^2 & 0 & 0 \\ 1/2T^2 & T & 0 & 0 \\ 0 & 0 & 1/3T^3 & 1/2T^2 \\ 0 & 0 & 1/2T^2 & T \end{bmatrix}$$

q is a scalar, and T is the sampling time. For the CV motion model, the $f^{(r_k)}(\cdot)$ function can be replaced by the transition matrix $F^{(r_k)}(\cdot)$. When r_k=1, $F^{(r_k)}(\cdot)$ corresponds to the standard CV model

$$F^{(1)}(\mathbf{x}_k)=\begin{bmatrix}1 & T & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & T\\ 0 & 0 & 0 & 1\end{bmatrix}$$

r_k=2,3 correspond to clockwise and anticlockwise CT motions, respectively.

$$F^{(j)}(\mathbf{x}_k)=\begin{bmatrix}1 & \frac{\sin(\omega_k^{(j)}T)}{\omega_k^{(j)}} & 0 & -\frac{1-\cos(\omega_k^{(j)}T)}{\omega_k^{(j)}}\\ 0 & \cos(\omega_k^{(j)}T) & 0 & -\sin(\omega_k^{(j)}T)\\ 0 & \frac{1-\cos(\omega_k^{(j)}T)}{\omega_k^{(j)}} & 1 & \frac{\sin(\omega_k^{(j)}T)}{\omega_k^{(j)}}\\ 0 & \sin(\omega_k^{(j)}T) & 0 & \cos(\omega_k^{(j)}T)\end{bmatrix}, j=2,3$$

Here the mode-conditioned turning rates are given by

$$\omega_k^{(2)}=\frac{a_m}{\sqrt{\dot{x}^2+\dot{y}^2}}$$

$$\omega_k^{(3)}=-\frac{a_m}{\sqrt{\dot{x}^2+\dot{y}^2}}$$

where a_m is the constant maneuver acceleration parameter.

B. Sensor Measurement Model

Three types of sensors are used in our work. These are 1) ESM sensor that reports bearing-only measurements, 2) range sensor that reports range measurements and 3) 2D RADAR sensor that reports range-bearing measurements [8]. The measurement model can be mathematically written as

$$\mathbf{z}_k^j=h^{(i)}(\mathbf{x}_k)+\mathbf{w}_k$$

where $\mathbf{z}_k^j$ is the measurement from sensor j. $h^{(i)}(\cdot)$ corresponds to three types of sensor measurement models, $i=1,2,3$

$$h^{(1)}(\mathbf{x}_k)=\tan^{-1}\left(\frac{y_k-y^{s_j}}{x_k-x^{s_j}}\right)$$

$$h^{(2)}(\mathbf{x}_k)=\sqrt{\left(y_k-y^{s_j}\right)^2+\left(x_k-x^{s_j}\right)^2}$$

$$h^{(3)}(\mathrm{x}_k) = \begin{bmatrix} \tan^{-1}\left(\dfrac{y_k - y^{s_j}}{x_k - x^{s_j}}\right) \\ \sqrt{(y_k - y^{s_j})^2 + (x_k - x^{s_j})^2} \end{bmatrix}$$

where (x_k, y_k) is the target position at time k, (x^{s_j}, y^{s_j}) is the position of sensor j. $\mathbf{w}_k$ denotes the measurement noise. In our work, we examine the glint noise as well as standard Gaussian noise [5]. Glint noise has a non-Gaussian distribution, and a mixture approach is widely used in modeling the non-Gaussian glint noise. In the proposed tracking algorithm, the glint noise is modeled by a Gaussian Mixture Model (GMM) with two components.

$$\mathrm{w}_k \sim \alpha_g N(0, \Sigma_1) + (1 - \alpha_g) N(0, \Sigma_2)$$

where $\alpha_g > 0.5$ is the glint probability, and $\Sigma_1 < \Sigma_2$. Note that when $\alpha_g = 1$, glint noise degenerates to standard Gaussian noise with zero mean and covariance matrix Σ_1.

IV. TRACKING ALGORITHMS

This section describes the recursive algorithms implemented for tracking a single target using EKF or particle filter techniques. Two of the algorithms are EKF-based and the other two are PF-based schemes. The algorithms considered are (i) EKF-IMM, (ii) PF-IMM, (iii) EKF-Glint Noise, (iv)PF-Glint Noise. All four algorithms are applicable to both single-sensor and multi-sensor scenarios.

A. Extended Kalman Filter

Extended Kalman filter is a minimum mean square error (MMSE) estimator based on the Taylor series expansion [1]. The mean $\bar{\mathbf{x}}_k$ and covariance P_k of the Gaussian approximation to the posterior distribution of the states can be derived as follows:

$$\bar{\mathbf{x}}_{k|k-1} = f(\bar{\mathbf{x}}_{k-1}, 0)$$
$$P_{k|k-1} = F_k P_{k-1} F_k^T + Q_k$$
$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1}$$
$$\bar{\mathbf{x}}_k = \bar{\mathbf{x}}_{k|k-1} + K_k (y_k - h(\bar{\mathbf{x}}_{k|k-1}, 0))$$
$$P_k = P_{k|k-1} - K_k H_k P_{k|k-1}$$

where K_k is the Kalman gain, Jacobians of the process model and measurement model are given by

$$F_k \overset{\Delta}{=} \left.\frac{\partial f(\mathrm{x}_k)}{\partial \mathrm{x}_k}\right|_{(\mathrm{x}_k = \bar{\mathrm{x}}_{k-1})} = F^{(r_k)}(\mathrm{x}_k)$$

$$H_k \overset{\Delta}{=} \left.\frac{\partial h(\mathrm{x}_k)}{\partial \mathrm{x}_k}\right|_{(\mathrm{x}_k = \bar{\mathrm{x}}_{k|k-1})}$$

$$= \begin{bmatrix} -\frac{y_k - y^s}{(y_k - y^s)^2 + (x_k - x^s)^2} & 0 & -\frac{x_k - x^s}{(y_k - y^s)^2 + (x_k - x^s)^2} & 0 \\ \frac{x_k - x^s}{\sqrt{(y_k - y^s)^2 + (x_k - x^s)^2}} & 0 & \frac{y_k - y^s}{\sqrt{(y_k - y^s)^2 + (x_k - x^s)^2}} & 0 \end{bmatrix}$$

For centralized measurement EKF fusion

$$H_k = [H_{k,s_1}^T, H_{k,s_2}^T, \cdots, H_{k,s_n}^T]^T$$

$$R_k = \begin{bmatrix} R_{k,s_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & R_{k,s_n} \end{bmatrix}$$

where H_{k,s_i} is the Jacobian of the measurement model for each sensor, and R_{k,s_i} is the covariance of the measurements model.

For the maneuvering target tracking problem, the IMM algorithm has been shown to be one of the most cost effective and simple approaches. At each calculation cycle, the IMM consists of three major steps: interaction (mixing), filtering and combination. At each time, the initial condition for the filter matched to a certain mode is obtained by mixing the state estimates of all the filters at the previous time under the assumption that this particular mode is in effect at the current time. This is followed by a regular filtering step, performed in parallel for each mode. Then a combination of the updated state estimates of all the filters yields the state estimate.

B. Particle Filtering

Particle filters represent the state probability density function approximately through a set of samples and implement Bayesian recursion directly on the samples instead of dealing with the exact analytical functional representations of the distributions [2, 3]. Tracking framework based on particle filtering will show better performance on nonlinear/non-Gaussian problems. The recursive Bayesian filtering paradigm provides the *a posteriori* PDF $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$ via the prediction and update recursions.

Prediction:

$$p(\mathbf{x}_k | \mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k | \mathbf{x}_{k-1}) p(\mathbf{x}_{k-1} | \mathbf{z}_{1:k-1}) d(\mathbf{x}_{k-1})$$

Update:

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})}$$

where the state $\mathbf{x}_k$ evolution is described in terms of the transition probability:

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{x}_{k-1}) &= \int p(\mathbf{x}_k|\mathbf{x}_{k-1},\mathbf{v}_{k-1})p(\mathbf{v}_{k-1}|\mathbf{x}_{k-1})d\mathbf{v}_{k-1} \\ &= \int \delta(\mathbf{x}_k - f_{k-1}(\mathbf{x}_{k-1},\mathbf{v}_{k-1}))p(\mathbf{v}_{k-1})d\mathbf{v}_{k-1} \end{aligned}$$

And how the given $\mathbf{x}_k$ fits the available measurement $\mathbf{z}_k$ is described as:

$$p(\mathbf{z}_k|\mathbf{x}_k) = \int \delta(\mathbf{z}_k - \mathbf{h}_k(\mathbf{x}_k,\mathbf{w}_k))p(\mathbf{w}_k)d\mathbf{w}_k$$

For maneuvering target tracking, the aim of the optimal filter is to sequentially estimate the unknown hybrid hidden state $\{\mathbf{x}_k, r_k\}$ given the observations $\{\mathbf{z}_{1:k}\}$. Applying Bayes' rule, the formulation of the recursion that updates $p(x_{0:k-1}, r_{1:k-1} \mid \mathbf{z}_{1:k-1})$ to $p(x_{0:k}, r_{1:k} \mid \mathbf{z}_{1:k})$ can be derived as

$$p(x_{0:k}, r_{1:k} \mid \mathbf{z}_{1:k}) = p(x_{0:k-1}, r_{1:k-1} \mid \mathbf{z}_{1:k-1}) \frac{p(\mathbf{z}_k \mid \mathbf{x}_{0:k}, r_{1:k}, \mathbf{z}_{1:k-1}) f(x_k \mid x_{k-1}, r_{k-1}) \pi_{r_{k-1}r_k}}{C}$$

where C is a constant. The basic idea for solving maneuvering target tracking using a particle filter is to decouple the hybrid estimation problem into a discrete part and a continuous part. We assume that sensor measurements are independent from each other. Here is the summary of the particle filter solution for the maneuvering target tracking problem in sensor networks.

- Generate samples of $r_k^{(i)}$ from an importance proposal distribution $\tilde{r}_k^{(i)} \sim \pi\{r_k \mid Z_{1:k}, r_{1:k-1}^{(i)}\}$, where $Z_{1:k}$ represents the measurements from all the sensors up to time k. Generate samples $\tilde{\mathbf{x}}_k^{(i)} \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, r_k^{(i)})$.
- Evaluate the importance weights

$$\tilde{\varpi}_k^{(i)} \propto \varpi_{k-1}^{(i)} \frac{p(\tilde{r}_k^{(i)} \mid \tilde{r}_{k-1}^{(i)})}{\pi(\tilde{r}_k^{(i)} \mid Z_{1:k}, \tilde{r}_{1:k-1}^{(i)})} \prod_{j=1}^{S_n} p(z_k^j \mid Z_{1:k-1}, r_{1:k}^{(i)})$$

- Normalize the weights

$$\varpi_k^{(i)} = \frac{\tilde{\varpi}_k^{(i)}}{\sum_j^N \tilde{\varpi}_k^{(j)}}$$

- Resampling: multiply/discard particles $\{r_k^{(i)}, i = 1,2,...,N\}$ with respect to high/low normalized importance weights $\varpi_k^{(i)}$ to obtain N samples $\{r_k^{(i)}, \mathbf{x}_k^{(i)}\}_{i=1}^N$
- MMSE of $\bar{\mathbf{x}}_k = \sum_{i=1}^N \varpi_k^{(i)} \tilde{\mathbf{x}}_k^{(i)}$

V. Simulation Results

We address the problem of tracking a maneuvering target in noise using multiple heterogeneous sensors. Fig 2 shows the tracking scenario. In our experiments, we use the dynamic state space model to generate the synthetic data, and 50 Monte Carlo computation simulations were carried out to evaluate the performance of the algorithms for each experiment. The position mean square error is defined as

$$MSE_k = \frac{1}{N}\sum_{n=1}^{N}\left[(x_k - \bar{x}_k)^2 + (y_k - \bar{y}_k)^2\right]$$

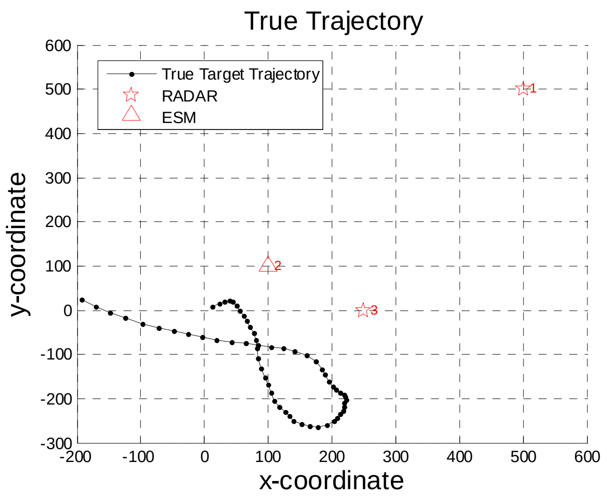


Fig 2. Simulated path of the target

We set sampling rate T=1, q=20, then

$$Q = \begin{bmatrix} 6.6667 & 10 & 0 & 0 \\ 10 & 20 & 0 & 0 \\ 0 & 0 & 6.6667 & 10 \\ 0 & 0 & 10 & 20 \end{bmatrix}$$

The typical maneuver acceleration parameter is set to α_m = 1m/s^2. Mode probability transition matrix used in the IMM is

$$\Pi = \begin{bmatrix} 0.9 & 0.05 & 0.05 \\ 0.6 & 0.3 & 0.1 \\ 0.6 & 0.1 & 0.3 \end{bmatrix}$$

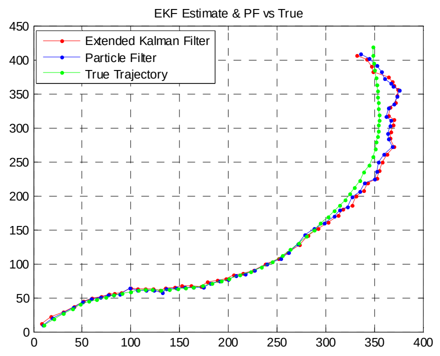
We set the glint probability $\alpha_g = 0.9$, measurement noise covariance $\Sigma_2 = 10\Sigma_1$, and

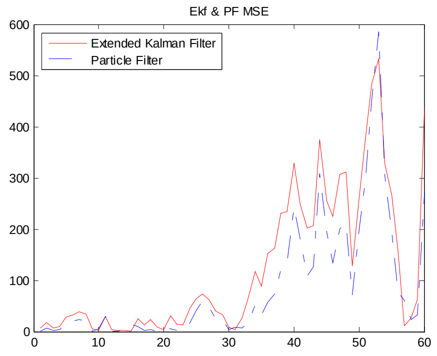
$$\Sigma_1 = \begin{bmatrix} 3.0462\text{e-}004 & 0 \\ 0 & 5 \end{bmatrix}$$

A. Tracking target using two bearing-only sensors

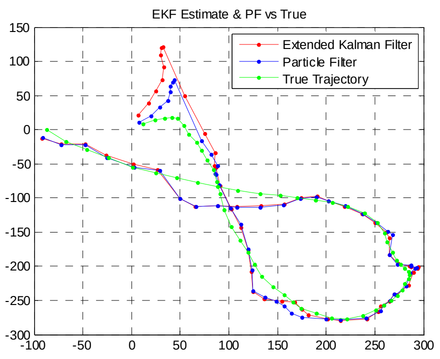
Bearing-only sensors are located at $(x^{s_1}, y^{s_1}) = (100m, 100m)$ and $(x^{s_2}, y^{s_2}) = (0m, 250m)$, respectively. PF shows better tracking performance than EKF. As we can see later, this is the most difficult case to track the target, and the tracking results of using two bearing-only sensors are much worse than those of using two range sensors. For the difficult case (A.2) where the target is maneuvering, we can see that the MSE for both EKF and PF are higher than those in the case where target motion follows a CV model (A.1).

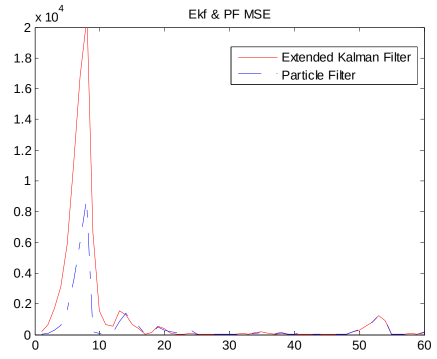
A.1 CV model with glint measurement noise





A.2 Maneuvering target

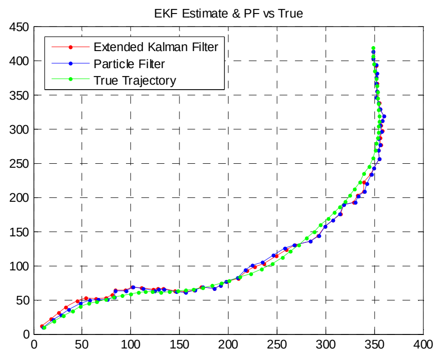


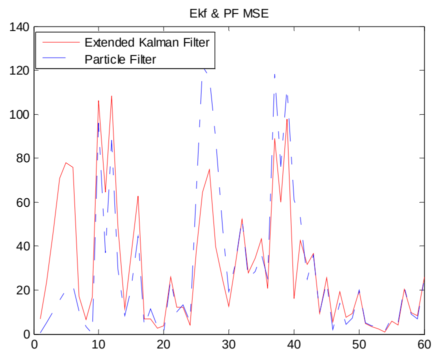


B. Tracking target using one radar sensor.

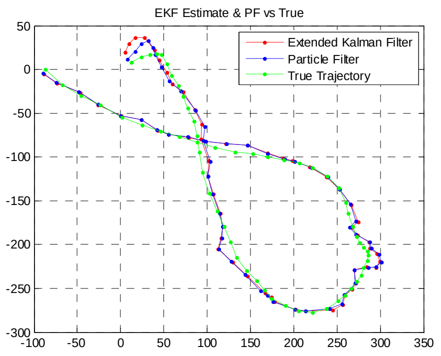
We assume that a radar is located at $(x^s, y^s) = (500m, 500m)$. From the experimental results, we can see that except for the first few steps, EKF and PF achieve almost the same MSE. But the computation time is much shorter for the EKF. For the difficult case where the target is maneuvering, we can see that the MSE for both EKF and PF are higher than those for the case of target motion that follows a CV model.

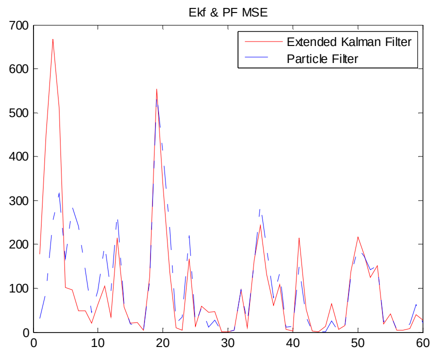
B.1 CV model with glint measurement noise





B.2 Maneuvering target

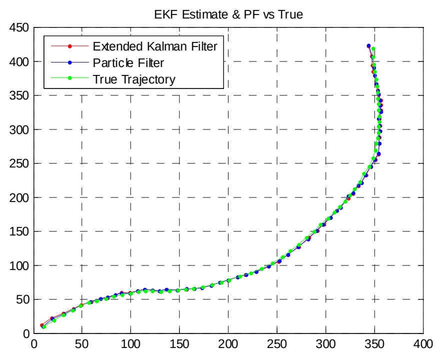


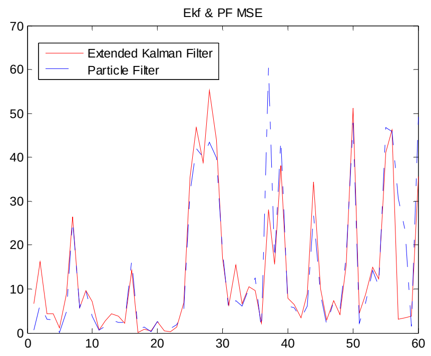


C. Tracking target using one range sensor and one bearing-only sensor

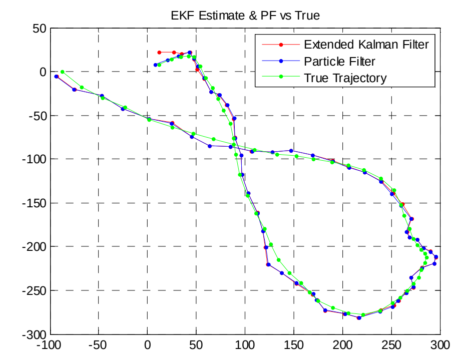
The range sensor position is set to $(x^{s_1}, y^{s_1}) = (0m, 250m)$, and bearing-only sensor position is set to $(x^{s_2}, y^{s_2}) = (100m, 100m)$. The function of bearing-only sensor plus range sensor is almost the same as a single radar sensor, except the bearing-only sensor and range sensor are located at different locations, so we have similar experimental results as in case B.

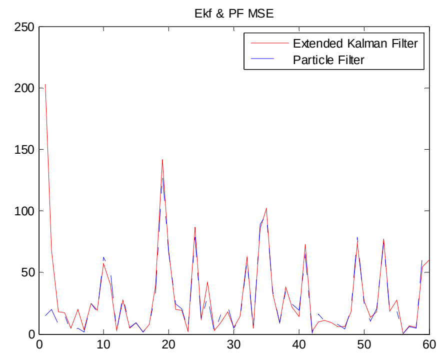
C.1 CV model with glint measurement noise





C.2 Maneuvering target

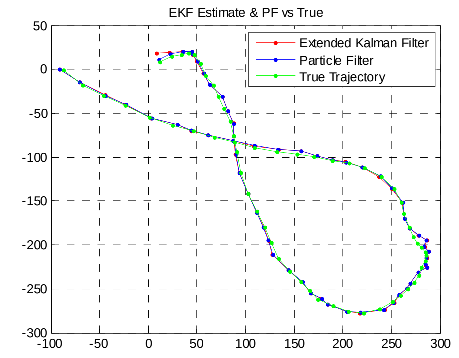


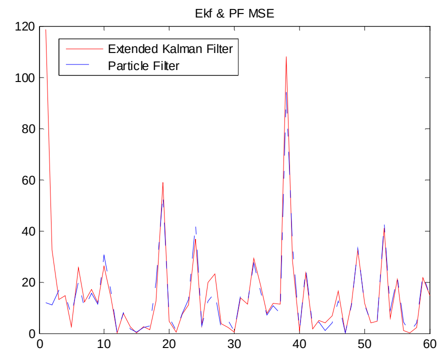


D. Tracking target using two range sensors

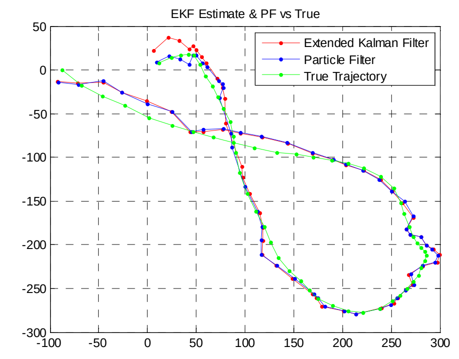
Two range sensors' positions are set to $(x^{s_1}, y^{s_1}) = (100m, 250m)$ and $(x^{s_2}, y^{s_2}) = (300m, 0m)$, respectively. For this sensor configuration, the PF still has similar steady-state performance as that of the EKF.

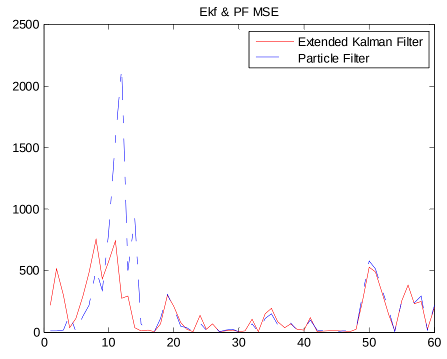
D.1 CV model with glint measurement noise





D.2 Maneuvering target

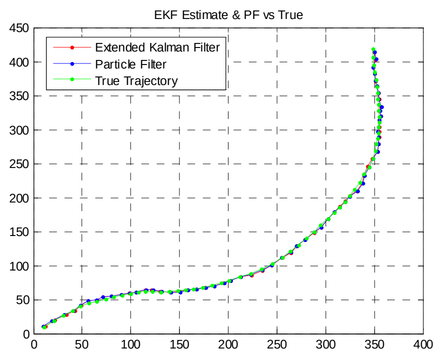


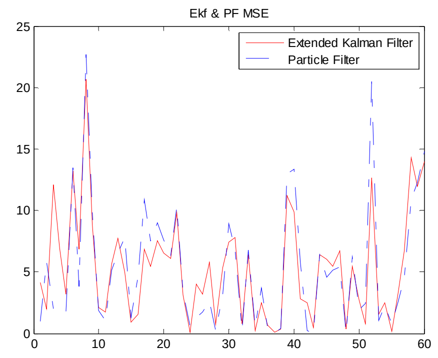


E. Tracking target using bearing only, range and radar sensors

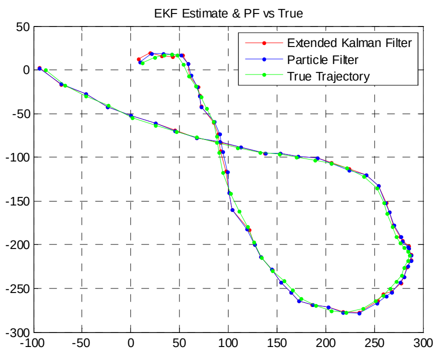
Sensors' positions are set to $(x^{s_1}, y^{s_1}) = (250m, 0m)$, $(x^{s_2}, y^{s_2}) = (0m, 250m)$ and $(x^{s_3}, y^{s_3}) = (500m, 500m)$. Here we use three different sensors, and MSE is much smaller than the previous cases. As expected, more accurate tracking results are achieved, since we are fusing data from more sources. Again, the PF and the EKF have very close steady-state performance.

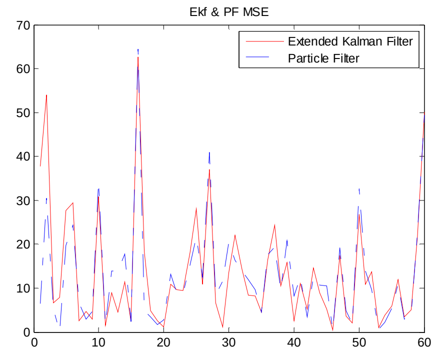
E.1 CV model with glint measurement noise





E.2 Maneuvering target





From the above experiments, we can observe that even when the measurement model is nonlinear for all types of sensors, using particle filtering does not always achieve a better steady-state performance in terms of MSE. Only under the conditions where only bearing-only sensors are used, the particle filter shows better performance than the EKF. We also found that even when the initial condition for both EKF and PF is the same, in the first few tracking steps, PF tracking results are more accurate than EKF. This is a valuable characteristic, especially when clutter and false alarms are among the measurements. When the measurements contain many false alarms, there is uncertainty as to which measurement is from the target and which is a false alarm. With such uncertainty, inaccurate estimates even at one time step could lead the filter to diverge and result in the loss of the target track. The PF has the potential to maintain the target track for a longer time in such harsh and realistic conditions. This issue needs further investigation in the future.

REFERENCES

[1] Y. Bar-Shalom and T.E. Fortmann, "Tracking and Data Association in Mathematics in Science and

Engineering Series," 179 Academic Press, San Diego, CA, 1988.

[2] N. Gordon, D. Salmond, and A. F. M. Smith, "Novel approach to nonlinear and non-Gaussian Bayesian state estimation," in *Proceedings IEE-F*, vol. 140, pp. 107--113, 1993.

[3] A. Doucet, N.de Freitas, and N.J. Gordon, Eds., "Sequential Monte Carlo Methods in Practice Series: Statistics for Engineering and Information Science," Springer-Verlag, New York, 2001.

[4] X. Rong Li, and V.P. Jilkov, "Survey of maneuvering target tracking part I: Dynamic models," in *IEEE Transactions on Aerospace and Electronic Systems* Vol. 39, 2003

[5] Bilik, I. Tabrikian, J. "Target tracking in glint noise environment using nonlinear non-Gaussian Kalman filter", Radar, 2006 IEEE Conference on, 24-27 April 2006

[6] M. Sanjeev Arulampalam, B. Ristic, N. Gordon and T. Mansell, "Bearings-Only Tracking of Manoeuvring Targets Using Particle Filters", EURASIP Journal on *Applied Signal Processing* 2004:15, 2351–2365

[7] Arnaud Doucet, N. Gordon, and Vikram Krishnamurthy, "Particle Filters for State Estimation of Jump Markov Linear Systems", IEEE TRANS. on *Signal Processing*, VOL. 49, NO. 3, MARCH 2001

[8] X. Rong Li and Vesselin P.Jilkov, "A Survey of Maneuvering Target Tracking—Part III: Measurement Models," *SPIE Conf. on Signal and Data Processing of Small Targets,* 2001

Appendix II. Summer Student Report 2007: Chris Poore

Particle Filters and Simulation Analysis

by

Christopher Poore

August 17, 2007

Introduction

During the summer of 2007 I worked in the Fusion Technology branch (IFEA). I was part of an in-house team focused on developing and improving non-linear tracking techniques. I aided in the finalization to the Development and Evaluation of Fusion Techniques program known as DEFT. I also worked on the In-House Particle Filtering and Testing program along with the newest effort: Tracking Evasive Non-Linear Targets (TENT).

These efforts in one way or another were related to non-linear tracking. I focused on developing and evaluating different techniques to deal with non-linear tracking. I researched a variety of filters based upon the Kalman filter and the particle filter. Other topics of interest I've examined include: out of sequence measurements (OOSMs), multiple target tracking, flight dynamics, sensor characteristics, methods for evaluating tracker performance, probability theory and statistics, as well as many others.

In this report a basic introduction to particle filters illustrating the details of its operation as well as its pros and cons are provided. Thorough analysis of in-house simulations run during the summer is presented. Two sources for code are presented for future reference. Finally recommendations for future research are detailed.

Background on Particle Filters

Kalman filters have been used for decades and various adaptations have been produced through the years to deal with troubling issues such as dealing with non-linearity. A solution used to address some of the weaknesses exhibited in Kalman filters is the particle filter. Although particle filtering has been around for quite some time too, it has only recently come back into the spotlight. In the past the particle filtering was deemed too computationally expensive for realistic use but with the advent of faster processors and technological upgrades, particle filters are acquiring a new look.

"Particle filters are sequential Monte Carlo methods based upon point mass (or 'particle') representations of probability densities, which can be applied to any state space model, and which generalize the traditional Kalman filtering methods" [1]. In other words, a particle filter can estimate a state space model by representing the posterior density function using many random samples (particles) with assigned weights. As the number of these particles increases the representation of the pdf reaches its optimal state.

There are numerous variations of particle filters but one of the most basic is a Sequential Importance Resampling (SIR) Filter. There are only a few basic steps to this filter. Initialization begins with establishing a prior probability distribution, that is, an initial guess in the form of a probability distribution over the state at the start time. This corresponds to how likely it is for the target to be at a given location. Next a set of particles are drawn from this initial prior probability distribution and the filter is initialized.

Once the filter is initialized it repeats three basic steps. The first is generating a proposal distribution. This step asks "where could the target have moved to given what I knew about where it might have been an instant before." It predicts the state of the target given all previous observations.

The next step is incorporating observations. This step takes information from the sensors and updates the belief where the target is located by assigning importance weights to all the particles.

The final step is resampling and it is required for the filter to maintain diversity. SIR filtering is unique in that the resampling step aids in reducing the degeneracy problem. After a few iterations of the basic particle filtering process all but one particle will have negligible weight. Resampling determines the number of effective particles (i.e. particles with high weights) remaining and if it is below a certain threshold shifts the negligible particles closer to the effective particles. The filter resamples particles according to the renormalized weights. Particles with higher weights get sampled more often than particles with lower weights.

The particle filter has several benefits compared to other filters. The particle filter performs well in non-linear situations and is capable of handling a non-linear state and observation model. The particle filter is also designed to perform well with non-Gaussian distributions and allows the use of multi-modal distributions. Unlike some filters the particle filter estimates the full probability density function of the state. With a near infinite number of particles the results produced are close to the optimal solutions.

There are also several drawbacks with using a particle filter. The most common concern is its computational cost. As the particles increase in number, the number of computations increases dramatically. Along with other filters the particle filter is also prone to the curse of dimensionality. As the number of dimensions increase the computational complexity increases exponentially. The problems associated with degeneracy mentioned previously exist in many versions of particle filters. Often particle filters simply aren't necessary in some applications and a basic linear-Gaussian assumption is sufficient to produce the desired results.

In-House Simulations

Utilizing in-house software already developed in collaboration with Syracuse University, I compared the effectiveness of an Extended Kalman filter (EKF) against a particle filter (PF). The simulations run via MATLAB tracked a moving ground target using an IMM-EKF and a MM-PF with bearings-only sensors. The path of the target could be modeled in a variety of ways to increase or decrease the degrees of linearity. Any number of bearings-only sensors could be placed around the target. I analyzed the simulation and determined how numerous parameters interacted with one another.

Comparison of the EKF vs. PF depends on several factors including: sensor position, process noise (determined by a scalar multiplier "q"), number of particles used in the particle filter, measurement noise, and glint noise.

The first scenario that was extensively analyzed consisted of the ground target initially heading 160 degrees from North at 70 mph. After 20 seconds, the target executed a Counterclockwise Coordinated Turn for 3 seconds, to establish a new course. The target then traveled with a Constant Velocity motion until it executed its final maneuver for 4 seconds, a Clockwise Coordinated Turn.

For this scenario, two bearings-only sensors were activated at a time and positioned at fixed points surrounding the moving target. To understand the effects of sensor positioning, these two sensors were repositioned in 15 different setups.

Sensor position can degrade overall tracking performance for both EKF and PF. Overall tracking performance refers to RMSE and variance. A high tracking performance is described as having low RMSE and low variance. Performance is degraded when the sensors are not lined up normal to the target track. When the sensors are lined up in parallel with the direction of motion of the target, performance is minimized.

In this scenario, overall tracking performance was at a minimum when: 1) the two sensors were positioned side by side on all four sides of the target's track; 2) the two sensors were positioned on opposite sides of the target track parallel with the target's motion. Alternatively, tracking performance was maximized when either: 1) one or more of the sensors were positioned close to the target track so at no time during the simulation was the sensor in parallel with the target's motion; 2) the two sensors were positioned at right angles on the outside of the target tack ensuring at least one sensor was normal to the target at all times.

In the two cases mentioned previously where performance is minimized, the EKF outperformed the PF in nearly every situation. When the bearings-only sensors are positioned in such a manner the particle filter has difficulty recognizing range and as a result may make a wrong turn, while the EKF does not experience this problem. This may be due to the specific implementation of each filter or perhaps may suggest the particle filter is more suitable with an addition of a range sensor such as radar. Despite the EKF's performance over the PF, it is important to remember the RMSE and the variance were still very large for both filters in this sensor setup.

When the sensors are positioned in the two cases where performance is maximized the results weren't as one-sided toward the EKF as when the sensors were positioned poorly. A noticeable trend emerged from this sensor setup. The PF outperformed the EKF whenever "q" was set to a negligibly low value no matter how many particles were used (50, 100, 500). This poor performance by the EKF with a low "q" value was found to be the direct result of an initialization issue. In this scenario the two filters initialized their tracks at a different position than the actual initial target location. This caused the EKF to lose the track immediately after the first few time steps because the low "q" value represented high confidence in the EKF's measurements when in fact the measurement was far away from the actual target location. The particle filter did not exhibit any such behavior with the varying process noise levels. Thus it can be seen that the EKF is highly dependent on process noise while the particle filter is less dependent.

When the PF was set to 50 particles, in this performance maximization sensor setup once again, the EKF showed better performance for the "q" values of 0.5, 1, and 5. As the number of particles increased, the performance of the PF increased (with the exception that simulation time increased dramatically as a result from more computational complexity). When the number of particles was increased to 100 the EKF outperformed or did just as well as the PF with the exception of when "q" was set to a negligibly small number such as 1.6×10^{-6} as mentioned above. Once the number of particles was set to 500 the PF either outperformed or did just as well as the EKF. As expected theoretically, as the number of particles approaches infinity, the closer the results will become optimal.

The effects of varying levels of measurement noise were briefly evaluated. For ranging levels of measurement noise it was discovered there exists a maximum and minimum threshold not to be exceeded. Performance deteriorates severely when the measurement noise becomes too high or even too low.

The ensuing series of scenarios generated focused on the degrees of non-linearity of the target's track. The main independent variable was track configuration. Four different paths for the ground target were examined: a straight line, a circle, a u-turn, and a multiple turn model. The parameters tested in the previous scenario: process noise, measurement noise, and number of particles were set to fixed values. Multiple bearings-only sensors were positioned in fixed locations around all four target tracks in an attempt to minimize the impending effects of sensor positioning discovered in the first scenario.

Results from these four different target paths failed in their attempt at providing insight on the effects of target non-linearity. For all four target paths the EKF and PF produced nearly identical results and no visible pattern could be established as to when one filter outperformed the other. However this may be due to a fundamental flaw in the simulation itself. Sampling from the measurements occurs at every time step and as a result may be sampling too quickly. With such a high measurement sample rate the two filters may be impersonating a non-linear trajectory such as a circle as linear.

Although increasing the number of sensors in these scenarios was intended to be a panacea for dealing with sensor alignment issues discussed in the first scenario, it failed in simplifying the experiments. With an increase in the number of sensors tracking performance is naturally increased. However with increased performance for both filters it becomes difficult to visually substantiate one filter's performance over the other. If the number of sensors is reduced, performance is reduced to the point where it is difficult to determine whether a specific filter reacting to non-linearity is to blame or whether it's just poor sensor positioning coupled with a high sampling rate.

The effect of glint noise was also examined for the multiple turn model of the target trajectory. It was clear that glint noise had a detrimental effect on performance. As the effects of glint noise were increased, performance decreased. Although the performance did not change immensely as the effects of glint were increased (perhaps due to the use of multiple sensors and high sample rate), the transient response of the generated tracks showed a moderate change in performance. More importantly, and perhaps of high note, the PF initially outperforms the EKF in the presence of glint. This may prove the theory that particle filters can outperform an EKF in non-Gaussian noise environments.

Other Examples of Code

In an undertaking to fully understand the efficacy of a particle filter, past implementations of particle filters already constructed were investigated. Two main sources are to be highlighted. The first is code written by Michael A. Goodrich to support Brad Huber's thesis work [2] and the second source is a series of examples from Dan Simon's book Optimal State Estimation: Kalman, H-infinity, and Nonlinear Approaches [3].

Huber's thesis proposed to equip a mini-UAV with radio sensors and use Bayesian filtering techniques to locate an Emergency Locator Transmitter for a downed aircraft. Huber created a MATLAB simulation of a mini-UAV with noisy non-linear sensor behavior that searches for a downed aircraft. Goodrich adapted Huber's code and created a simplified version of the existing particle filter implementation.

The code is important because it acts as an analysis tool. One can directly see the effects of changing particle filter related parameters on the simulation. Since most of the information available on particle filtering is theory and pseudo code, the code provides a rare opportunity for seeing actual values.

The code from Dan Simon provides several examples of not only particle filtering but other examples of non-linear, Kalman, and H-infinity filtering. Simon's examples are basic and designed to be simple supplemental guides to his book. Studying these examples provide all the same advantages as Huber's thesis work but also offer variety and alternative techniques.

Recommendations for Future Research

While understanding that some initial research may have already taken place and that it takes time to produce results, recommendations for future research are still presented.

When trying to produce a simulation it is absolutely essential that the confines and parameters are known. For the future, work needs to be done researching "real world" values to apply to simulations. Data needs to be collected using existing implementations of non-linear tracking as well as sensor information. Research should not only include reading for information but cataloging the results of what has been learned.

Development of a sophisticated scenario generator needs to be developed. A generator that is capable of handling multiple tracking filters with an easy to use interface for changing parameters would be the ultimate goal. The tracks and values generated should be based on realistic recorded data. Methods to evaluate tracking performance such as root mean square error, variance, or Cramer-Rao Lower Bounds should be included.

If future work on the existing software is desired, changes should be immediately made to the measurement sample rate. If new results are generated from that change one may be able to see the effects from varying parameters more clearly. Introduction of alternative types of sensors such as radar may show different results. Further research into the effects of non-Gaussian noise on different filters should be looked into.

It is necessary to establish a general base of knowledge before advanced measures can be introduced. While remembering to take a "crawl before you walk" approach, future work in implementing advanced measures should be implemented after preliminary results have been produced. Meaning it is important to understand how the elements of non-linear tracking interact with one another at the fundamental level first and foremost. Once this is achieved more complex incorporations may be introduced to simulations such as Out of Sequence Measurements (OOSMs), multiple target tracking, different filter types, adaptive filters, smoothing techniques, or any others.

References

[1] Arulampalam, Maskell, Gordon, Clapp: "A Tutorial on Particle Filters for on-line Non-linear / Non-Gaussian Bayesian Tracking," IEEE Transactions on Signal Processing, Vol. 50, 2002.

[2] *Particle Filter Code.* Retrieved July 2007 from BYU HCMI PBwiki: http://hcmilab.pbwiki.com/

[3] Simon, Dan. Optimal State Estimation: Kalman, H-infinity, and Nonlinear Approaches. John Wiley & Sons, 2006.

Appendix III. Summer Student Report 2007: Becky Bailey

Nonlinear Filtering Using Solutions to the Fokker-Planck Differential Equation

Target tracking has been one of the problems encountered by engineers for many decades. The main goal of this problem is to track a target that is moving on some trajectory using measurements given by sensors. Although on the surface, this may seem like a fairly simple problem, there are numerous challenges contained within the different tracking scenarios. These include the facts that sensors produce noisy measurements, data is often received at irregular time intervals, obstructions occurring in the "view" of the target (i.e. bridges, buildings, etc.), data association problems due to multiple targets, and nonlinearity of tracks. Due to the previously mentioned challenges that occur in target tracking, developing an algorithm that solves the entire tracking problem *and* takes into account the many different scenarios that can develop has been a very difficult task for engineers. This paper will focus on one of the main problems in tracking, which is the nonlinearity of a target's track. More specifically, a new nonlinear tracking filter will be explained which uses solutions of the Fokker-Planck differential equation to solve the nonlinear tracking problem.

In general, tracking algorithms can be divided into three main steps: initialization, prediction, and update. All tracking filters that have been developed deal with these three steps in some form or another. The initialization step refers to the process of obtaining an initial state based on a certain degree of uncertainty. The prediction step predicts where the target should be based only on previous measurements using the system dynamical model. This creates the state estimate. The update step actually uses the measurement upon arrival from the sensor to update the state estimate. A probabilistic based weighting is used to create an optimal combination between old and new information. [1]

The majority of filtering algorithms developed up to this point have either dealt with linear tracks, or used some sort of linearization process to linearize the nonlinear target tracks. One of the most widely known filtering algorithms developed for target tracking is the Kalman Filter, which was developed in the 1960's by Rudolf Kalman. Although this filter is the optimal filter for the linear-Gaussian tracking problem, its real world application is minimal due to the numerous assumptions made in the theory. In particular, the Kalman Filter not only assumes the target track to be linear, but the noise signal and the posterior density are both assumed to be Gaussian. These are very limiting assumptions because rarely does this type of scenario occur in the real world, making it obvious that the Kalman Filter needed to be extended to nonlinear situations. This was done with the Extended Kalman Filter (EKF), which basically linearized the nonlinear situations by assuming local linearity of the target's track.

The filters discussed thus far have not really "solved" the nonlinear tracking problem, but have linearized the nonlinear problem, and then calculated a linear solution. While there have been some other nonlinear tracking filters that do not actually solve the Fokker-Planck Equation (FPE), such as the particle filter, my focus during my time here has been on nonlinear filters that solve the Fokker-Planck Equation. To understand these filters, and their approach to the nonlinear tracking problem, one must first have an understanding of what the FPE is and how it is used in tracking.

The Fokker-Planck Equation is a second order partial differential equation of motion for the probability density function of continuously changing macroscopic variables. Solving this equation leads to a probability density function enabling one to find any of the averages for macroscopic variables by integration. When applied to nonlinear filtering, this equation explains the progression of the conditional probability density of the state vector between the times at which measurements are actually taken. To solve this equation in real time would solve the nonlinear filtering problem. The difficulty in this is that the solution suffers from the "curse of dimensionality," and therefore the computational complexity of the solution grows exponentially with the dimension of the state vector. [2]

The general FPE is given in the following form:

$$\frac{\partial W}{\partial t} = \left[-\sum_{i=1}^{N} \frac{\partial}{\partial x_i} D_i^{(1)}(\{x\}) + \sum_{i,j=1}^{N} \frac{\partial^2}{\partial x_i \partial x_j} D_{ij}^{(2)}(\{x\}) \right] W(x,t) \qquad (1)$$

With the variables being defined as follows:
$D_i^{(1)}$ = Drift vector
$D_{ij}^{(2)}$ = Diffusion tensor
$W(\{x\},t)$ = Distribution function

The FPE is actually defined by the first 2 terms of the Kramers-Moyal expansion. The reason that it is only the first two terms of the Kramers-Moyal expansion is that the expansion with an infinite number of terms stops after the second term due to the fact that the coefficients vanish for $n \geq 3$, according to the Langevin equations from which they are derived. The drift vector and diffusion tensor are therefore just the first two coefficients of the Kramers-Moyal expansion.

In nonlinear filtering, the FPE takes a specific form based on the model used to define the dynamic system. This model is given by the following equation:

$$\dot{x}' = f(x', t) + \omega \tag{2}$$

Where x' is the system state vector, while the forcing function, ω is a random process. The probability density function $p(x,t)$ for the state variables defined by (2) satisfies the FPE of the following form:

$$\frac{\partial p}{\partial t} = -\left(\frac{\partial p}{\partial x'}\right) f - p\left[tr\left(\frac{\partial f}{\partial x'}\right)\right] + \frac{1}{2} tr\left(Q \frac{\partial^2 p}{\partial x'^2}\right) \tag{3}$$

In this case, $\frac{\partial p}{\partial x'}$ is the gradient of p with respect to x', $\frac{\partial f}{\partial x'}$ is the Jacobian of f with respect to x', $\frac{\partial^2 p}{\partial x'^2}$ is the Jacobian of the transpose of p with respect to x', and Q is the process noise matrix. [3]

In his paper [4], Fred Daum develops a filter which solves the FPE, and in combination with an exponential function, he solves the nonlinear filtering problem. Unlike the Kalman filter which can only handle the class of Gaussian probability densities, Daum's filter generalizes this so that any multivariate probability density from the exponential family can be used. In this theory, Daum considers the random variable $x(t)$, which evolves continuously in time according to the following equation:

$$dx(t) = f(x, t)dt + G(t)dw \tag{4}$$

In this case, f is a nonlinear function of x, and $w(t)$ is an R^n-valued Brownian motion. He then assumes that there exists a positive real-valued function, $\Psi = \Psi(x,t)$ that satisfies the following FPE corresponding to (4):

$$\frac{\partial p}{\partial t} = -\left(\frac{\partial \psi}{\partial x}\right) f - \psi\left[tr\left(\frac{\partial f}{\partial x}\right)\right] + \frac{1}{2} tr\left(Q \frac{\partial^2 \psi}{\partial x^2}\right) \tag{5}$$

With the initial condition:

$$\psi(x,t_0) = \left[p(x,t_0) e^{\{(\frac{1}{2})(x-m_0)^T P_0^{-1}(x-m_0)\}} \right]^{\frac{1}{s}} \tag{6}$$

If all of Daum's assumptions and conditions are satisfied, the result of his theory is an exact formula for the unnormalized probability density of the state at a specific time conditioned on the set of discrete time measurements $Z_k=\{z_1, z_2, \ldots, z_k\}$. This unnormalized probability density is given by the following formula:

$$p(x_k, t_k | Z_k) = \psi^s(x_k, t_k) \cdot \exp\left[-\left(\tfrac{1}{2}\right)(x_k - m_k)^T P_k^{-1} (x_k - m_k)\right] \tag{7}$$

Where m_k and P_k can be computed recursively using the following formulas:

$$\begin{aligned} m_k &= \overline{m}_k + P_k H_k^T R_k^{-1}(z_k - H_k \overline{m}_k) \\ P_k &= \overline{P}_k - \overline{P}_k H_k^T (H_k \overline{P}_k H_k^T + R_k)^{-1} H_k \overline{P}_k \end{aligned} \tag{8}$$

Although the recursive formulas for m_k and P_k lead the reader to believe that they are the mean and covariance of the state, they are not. In Remark 3 of his paper, Daum states, "Note that m_k is *not* the conditional mean of x_k, and P_k is *not* the conditional covariance matrix of x_k. But rather, m_k and P_k can be thought of as the mean and covariance matrix of some R^n-valued auxiliary Gaussian random variable." [4] Since it is very difficult to create a recursive filter without the mean and covariance, in [3] Schmidt develops a way to relate Daum's variables m_k and P_k to the actual mean and covariance $\hat{x}$ and M. The following equations were derived by Schmidt for the mean and covariance based on Daum's theory:

$$\begin{aligned} &\dot{M} = BM + MB^T - M(2L + g_1 G_1 + \ldots + g_n G_n)M + Q \\ &\text{AND} \\ &\dot{\hat{x}} = \hat{n} + B\hat{x} - M(2L + g_1 G_1 + \ldots + g_n G_n)\hat{x} - MS \end{aligned} \tag{9}$$

As a reader familiar with tracking can quickly see, the equation for propagating the covariance only differs from that of the EKF by the factor of

$$M(2L+g_1G_1+...+g_nG_n)M \tag{10}$$

It is also evident that the equation for propagating the mean only differs from that of the EKF by the factor of

$$M(2L+g_1\ G_1+...+g_nG_n)\hat{x} \tag{11}$$

These two factors, (10) and (11), are the part of the propagation equations that take into account Daum's parameters m_k and P_k. The update equations are also very similar to those of the EKF and are given as follows:

$$\begin{aligned}&M_k(+)^{-1}\hat{x}_k(+)=H^TR^{-1}z_k+M_k(-)^{-1}\hat{x}_k(-)\\&AND\\&M_k(+)^{-1}=H^TR^{-1}H+M_k(-)^{-1}\end{aligned} \tag{12}$$

Where the parameters before update are indicated by (-), and after update they are indicated by (+).

Overall, Daum was able to divide the tracking estimation problem into two separate steps. One step consists of the recursive calculation of m_k and P_k based on the sensor measurements. As previously revealed, this can be done with equations very similar to those of the Kalman filter. Schmidt took this step one level further, and developed propagation and update equations for the mean and covariance based on Daum's parameters m_k and P_k. The other general step in Daum's theory consists of the calculation of $\Psi(x,t)$, which does *not* depend on the sensor measurements. This calculation can be done off-line, before the actual observation interval using numerical solutions to the FPE.

Daum's filter that was previously described is a "finite-dimensional" filter, which means that it can be implemented by integrating a given number of ordinary differential equations. [4] In [5], Daum, along with Lambert and Weatherwax, develops a different filter, which they call the "wave filter." This algorithm solves the nonlinear tracking problem by simply solving the FPE for the conditional probability density function. This is done in a "split-step solution," which suffers from the "curse of dimensionality," unlike Daum's previously explained filter. The idea behind this filter is to solve the FPE using the "split-step solution," which can be viewed as the propagation step of the algorithm. Then for the update step, the conditional density is updated upon arrival of new measurements using Bayes' rule. This is given by:

$$p\left(x_{t_k} \middle| Z_k\right) = \frac{p\left(z_k \middle| x_{t_k}\right) p\left(x_{t_k} \middle| Z_{k-1}\right)}{p\left(z_k \middle| Z_{k-1}\right)} \tag{13}$$

In constructing the "split-step solution," the following system dynamic model was used to describe the dynamic behavior of the system:

$$dx_t = f(x_t) + dw_t \tag{14}$$

In this model, the process noise vector w_t is a zero mean Gaussian white noise process with spectral density matrix Q_t [5]. The FPE used to describe the evolution of the conditional probability density between measurements is given by:

$$\frac{\partial p}{\partial t} = -f_t^T \frac{\partial p}{\partial x_t} - tr\left(\frac{\partial f_t}{\partial x_t}\right) p + \frac{1}{2} tr\left(Q_t \frac{\partial^2 p}{\partial x_t^2}\right) \tag{15}$$

The solution described here will consider two separate special cases of this FPE in order to come to one unified solution. The first case considered will be the case where there is time invariant process noise, and therefore, $f_t = 0$ and $Q_t = Q$. With these conditions in mind, the FPE reduces to:

$$\frac{\partial p}{\partial t} = \frac{1}{2} tr\left(Q \frac{\partial^2 p}{\partial x_t^2} \right) \tag{16}$$

This reduced form of the FPE can actually be viewed as the diffusion equation with Q being the diffusion coefficient. Equation (15) can be solved by computing the convolution integral using an algorithm such as the Fast Fourier Transform (FFT). The second special case considered in this split-step solution is the case where there is no process noise, and therefore $Q_t = 0$. In this case, the FPE reduces to:

$$\frac{\partial p}{\partial t} + f_t^T \frac{\partial p}{\partial x_t} = -tr\left(\frac{\partial f_t}{\partial x_t} \right) p \tag{17}$$

In this form, f_t can be viewed as the drift vector, while the equation itself has actually been reduced to the convection equation. With a bit of manipulation, (16) can be transformed into a system of ordinary differential equations given by:

$$\frac{d}{dt}\begin{bmatrix} x_t \\ p \end{bmatrix} = \begin{bmatrix} f_t \\ -tr\left(\frac{\partial f_t}{\partial x_t} \right) p \end{bmatrix} \tag{18}$$

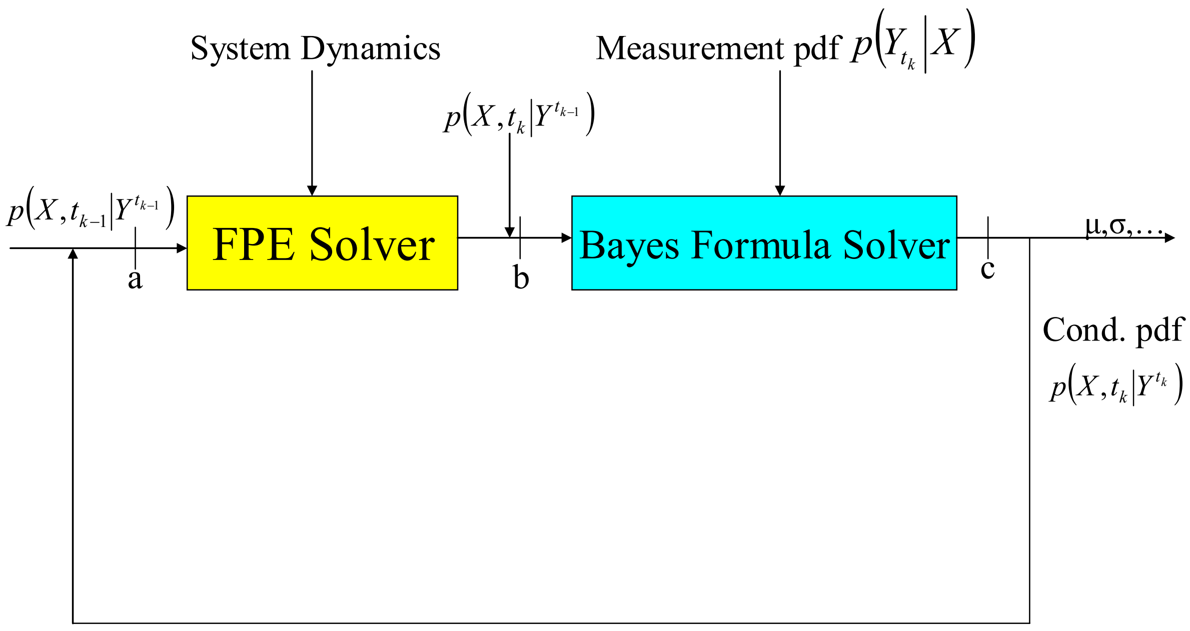
In order to solve (17), a grid must first be formed for the solution domain. Then, the state vector that corresponds to a particular grid point and the conditional density that goes along with it are propagated at the same time. Since these are ordinary differential equations, this can simply be done by using an ordinary differential equation solver, such as the 4th-order Runge Kutta integrator. [5]

By combining the two special cases, one can solve the general FPE used for this nonlinear filtering problem. This is done by requiring the process noise spectral density matrix to be discrete time. In essence, the situation being viewed here is the case where process noise is "injected" into the system at discrete time intervals, which coincide with the measurement times. In mathematical form, this means:

$$Q_t = \sum_k Q_k \delta(t - t_k) \tag{19}$$

Where $\delta(t-t_k)$ is the Dirac Delta function. By letting $Q_t = \sum_k Q_k \delta(t-t_k)$ one can say that immediately after the measurement step, the FPE reduces to the diffusion equation (15), and between measurements the FPE reduces to the convection equation (16). Since the solutions for both special cases have been described, the FPE for this nonlinear filtering algorithm can be solved. [5]

This technique of using numerical solutions of the Fokker-Planck equation to solve the nonlinear tracking problem can be simply viewed in the following flow chart:



http://ieeexplore.ieee.org/iel5/7/17885/00826335.pdf

[6]

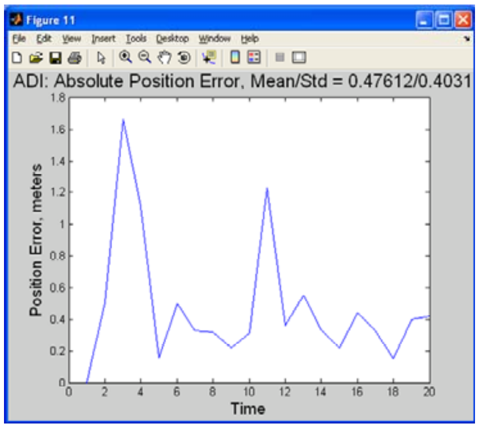
One of the other areas that I looked at this summer was the TENET (Techniques for Nonlinear Estimation of Tracks) software developed by Musick and Greenewald. [7] This piece of software implemented two tracking algorithms. One was a particle filter method, while the other used an ADI (Alternating Direction Implicit) finite difference method to solve the FPE. My focus was more on the ADI finite difference method, since it used a numerical approximation of the FPE to solve the tracking problem. The FPE it solved, defined by sub-operators, is given in the following form:

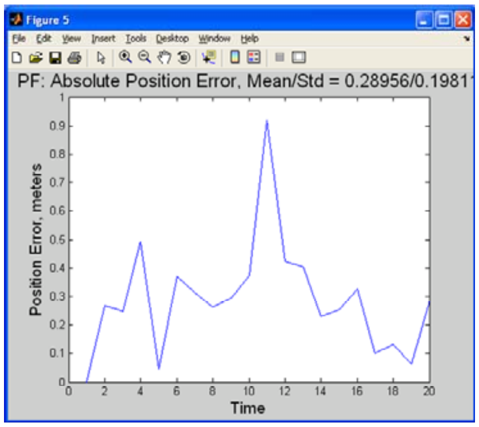
$$\frac{\partial p}{\partial t} = \sum_i A_i p$$

Where:

$$A_1 = -x\frac{\partial}{\partial x}$$
$$A_2 = \frac{q}{2}\frac{\partial^2}{\partial x^2}$$
$$A_3 = -y\frac{\partial}{\partial y}$$
$$A_4 = \frac{q}{2}\frac{\partial^2}{\partial y^2}$$
$$p = \text{continuum solution to the FPE}$$

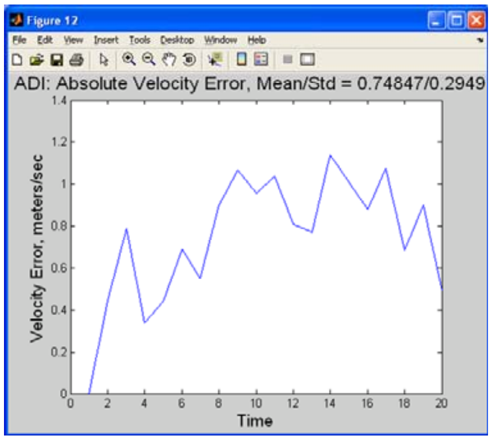
An alternating direction implicit scheme with finite difference methods is used to solve the FPE in this software. After running the software with one Monte Carlo run, my results are given in the following diagrams:

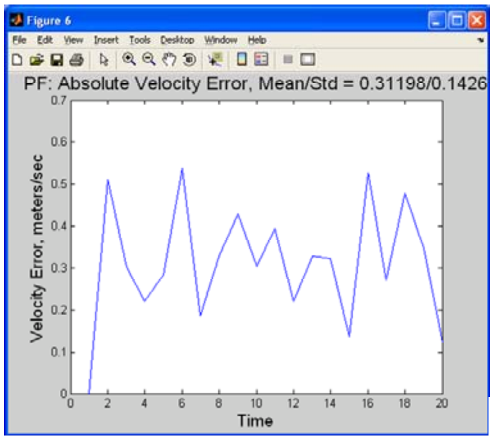




FPE Method PF Method

**Note: Y-Axis scales are different
[7]





FPE Method PF Method
** Note: Y-Axis scales are different
[7]

By comparing the error in the particle method against that of the FPE based method (ADI method), the following table was developed:

Error	FPE	PF
Position Error	.476 meters	.290 meters
Velocity Error	.748 m/s	.312 m/s

As one can easily see, the particle filter outperformed the FPE based method in both position and velocity error. This can be explained by the scenario tested in the TENET simulator. In this simulation, a single dim target was tracked in 2D space with a high degree of nonlinearity. The a high degree on nonlinearity and presence of only a single target creates a situation favoring the algorithm in the particle filter over that of the FPE based method. In the future, more work could be done with the TENET software, especially in looking into defining situations where the FPE based method outperforms the particle filtering method, computation of flops in order to compare which algorithm is more efficient, and comparing the performance of each algorithm against the resources used up in the running of the algorithm.

Designing nonlinear tracking filters has been a very difficult task for engineers. The most direct way to solve the problem seems to be by using numerical solutions to solve the Fokker-Planck equation for the conditional probability density, but this quickly becomes unreasonable due to the fact that the solution suffers from the "curse of dimensionality." Daum developed a filter which uses the Fokker-Planck equation for probability density in combination with an exponential function and sufficient statistics to solve the nonlinear tracking problem, but his conditions and assumptions are still fairly limiting to the real world application of the theory. Hopefully, in the future, a nonlinear tracking filter will be developed that defeats the “curse of dimensionality,” and is able to be applied to a wider range of real world scenarios.

References

[1] Simo Sarkka, Toni Tamminen, Aki Vehtari, and Jouko lampinen (2003). Probabilistic Methods in Multiple Target Tracking- Review and Bibliography. Published as technical report B36, ISBN 951-22-6938-4, Helsinki University of Technology. Laboratory of Computational Engineering, 2004.

[2] Risken, Hannes (1989). The Fokker-Planck Equation: Methods of Solution and Applications, 2nd ed., Springer-Verlag, Berlin.

[3] Schmidt, Garfield C. "Designing Nonlinear Filters Based on Daum's Theory." AIAA Journal of Guidance, Control, and Dynamics. vol. 16, no. 2, pp. 371-376, 1993 March-April.

[4] Daum, Frederick E. "Exact Finite-Dimensional Nonlinear Filters." IEEE Transactions on Automatic Control. vol. AC-31, no 7, pp. 616-622, 1986 July.

[5] Daum, F. E.; Lambert, H. C.; Weatherwax, J. L., "A Split-Step Solution of the Fokker-Planck Equation for the Conditional Density," Signals, Systems and Computers, 2006. ACSSC '06. Fortieth Asilomar Conference on , pp.2014-2018, Oct.-Nov. 2006.

[6] S. Challa and Y. Bar-Shalom. "Nonlinear Filter Design Using Fokker-Planck-Kolmogorov Probability Density Evolutions." IEEE Trans. Aero. Elect. Syst.,vol.36, no.1, pp.309-315, 2000. Available from: http://ieeexplore.ieee.org/iel5/7/17885/008 26335.pdf

[7] Greenewald, John, & Musick, Stanton. TENET Products, https://www.vdl.afrl.af.mil/programs/tenet/. July 2001.

List of Acronyms

AFRL/IF	Air Force Research Laboratory / Information Directorate
ADI	Alternating Direction Implicit
AIAA	American Institute of Aeronautics and Astronautics
ANTS	Adaptive Nonlinear Tracking System
BRAT	Baseline Road Assisted Tracker
BYU	Brigham Young University
C	constant
CAFÉ	Complementary Advanced Fusion Exploration
C2ISR	Command Control Intelligence Surveillance & Reconnaissance
CT	Constant Turn
CV	Constant Velocity
DEFT	Development and Evaluation of Fusion Techniques
DoN	Degree of Nonlinearity
EKF	Extended Kalman Filter
EKF-IMM	Extended Kalman Filter- Interacting Multiple Model
ELINT	Electronic Intelligence
ESM	Electronic Support Measures
FPE	Fokker Planck Equation
FTNLF	Fusion Techniques and Nonlinear Filtering
GMM	Gaussian Mixture Model
HPCs	High Performance Computers
IEE	Institute of Electrical Engineers
IEEE	Institute of Electrical and Electronics Engineers
IFTC	Information Directorate Computing Branch
IHPFAT	In-House Particle Filtering and Testing
IMINT	Image Intelligence
IMM	Interacting Multiple Model
IMM-EKF	Interacting Multiple Model Extended Kalman Filter
ISR	Intelligence Surveillance and Reconnaissance
JMS	Jump Markov System
KF	Kalman Filter
m	meters
MATLAB	Matrix Library (Trademark)
MC	Monte Carlo
MMSE	Minimum Mean Square Error
MM-PF	Multiple Model Particle Filter
MoN	Measures of Nonlinearity
MSE	Mean Square Error
MULTI-INT	Multiple Intelligence
N	Number of
NNGP	Non-Linear Non-Gaussian Processes
OOSMs	Out of Sequence Measurements
OOSP	Out of Sequence Problem
OWN	Ownship

OWN MVMT	Ownship Movement
PDF	Probability Distribution Function
pdf	Probability Density Function
PF	Particle Filter
PF-IMM	Particle Filter Interacting Multiple Model
q	Maneuver Index
RADAR	Radio Detection and Ranging
RF	Radio Frequency
RMS	Root Mean Square
RMSE	Root Mean Square Error
R^n	N-Dimensional ; Real Number Space
SIS	Sequential Importance Sampling
SIR	Sequential Importance Resampling
SMC	Sequential Monte Carlo
SPIE	Society of Photo-Optical & Industrial Engineers
SU	Syracuse University
T	Time
TENET	Techniques for Nonlinear Estimation of Tracks
TENT	Tracking Evasive Nonlinear Targets
tr	Trace
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter
UPF	Unscented Particle Filter
UTC	Universal Time Code
v	versus
z	Measurement
x	Track
2D	Two Dimensional